UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

VALENCIA

TRABAJO FIN DE GRADO EN
INGENIERÍA AEROESPACIAL

“Optimización de plataformas y puertas de embarque en aeropuertos”

Autor:
SERGIO LLOPIS JUAN

Tutor:
JUAN ANTONIO VILA CARBÓ

Escola Tècnica Superior d’Enginyeria del Disseny

JULIO DE 2021

Resumen

La optimización de las instalaciones aeroportuarias es un factor de vital importancia para la sostenibilidad económica de la aviación comercial. Una correcta gestión de los tiempos de espera de las aeronaves puede reducir considerablemente los costes para operadores y aerolíneas. El trabajo final de grado propuesto pretende recopilar las técnicas utilizadas actualmente en la gestión de puertas de embarque de los aeropuertos atendiendo a las necesidades actuales. Los estudios realizados se dividen en dos tipos según la naturaleza de los datos utilizados: determinista y probabilista. Siguiendo esta clasificación, se tiene como objetivo mostrar una versión simplificada del proceso de asignación mediante el desarrollo de un modelo de programación lineal propio para cada tipo de clasificación. Por último, se muestra una comparativa de las asignaciones estudiadas que permite observar que la elección de uno u otro obedecerá a las características particulares de cada aeropuerto, no pudiendo establecer conclusiones absolutas sobre la conveniencia de uno u otro.

Abstract

The optimization of the airport resources is key for the economic sustainability of the commercial aviation sector. The costs for airport operators and airlines can be importantly reduced by a precise managing of turnover times. The final degree project pretends to show the techniques used nowadays on boarding gates assignment while considering the actual needs. The research done on this field can de divided into two groups attending to the nature of the data used: deterministic and probabilistic. Following this classification, the objective of the project is to show a simplified version of the assignment process by means of one model for each type. Lastly, a comparison among the studied assignments is done, and it shows that the decision of using one or the other may be due to the particular characteristics of each airport, so no absolute conclusions can be extracted.

Agradecimientos

En este trabajo quiero expresar mi agradecimiento a todos los profesores del grado que me han permitido formarme profesionalmente en el ámbito de la ingeniería aeroespacial. En especial quiero agradecer la dedicación del tutor Joan Vila por su ayuda y su tiempo, imprescindibles para superar las dificultades surgidas en el proyecto.

Índice

Lista de tablas

Lista de ilustraciones

Lista de imágenes

Capítulo 1
Motivación

Como es bien conocido, el principal motivo para la elección del avión como forma de transporte es el tiempo de viaje ya que es el único sistema que permite recorrer distancias enormes en intervalos de tiempo asumibles por los pasajeros o ciertas mercancías. No obstante, el sector aeronáutico es uno de los más complejos a nivel financiero ya que cualquier operación que se realice implica una importante inversión previa. Si a esto se le añade la feroz competencia entre aerolíneas, la cantidad de normativa exigible y el gran equipo humano necesario, es fácil darse cuenta de que esta inversión viene acompañada de un margen de beneficio mínimo. No obstante, pese al enorme coste de la operativa, los billetes se mantienen ajustados con precios al alcance de la mayoría. Evidentemente el secreto detrás de unos precios tan ajustados no es otro que la utilización eficiente de los recursos, pues muchos de los costes (personal, mantenimiento, tasas…) se reparten entre los pasajeros de los múltiples vuelos que realiza una aeronave en un día. Así, no es de extrañar que una de las premisas básicas en el estudio financiero del sector sea la de mantener el mínimo tiempo posible los aviones en tierra ya que mientras están parados no generan beneficios.

El tiempo que estén los aviones en tierra dependerá en gran medida de la gestión realizada en el aeropuerto y de la facilidad que se encuentren los pasajeros a la hora de embarcar o desembarcar. Los tiempos son especialmente importantes si se tiene en cuenta que los aviones deben realizar múltiples servicios en un día. Como se ha comentado, se intenta ajustar al mínimo el tiempo que pasa un avión en el aeropuerto, pero esto implica que cualquier retraso en una de las paradas del día se irá acumulando a los posteriores vuelos. Además, se debe tener en cuenta que muchas aerolíneas ofrecen billetes entre destinos poco demandados mediante la estrategia *Hub and Spoke* (concentrar todos los vuelos en un mismo aeropuerto y conectar a los pasajeros mediante transbordos). Para que esta estrategia pueda funcionar se necesita que la sincronización sea máxima y que los pasajeros y sus equipajes se puedan desplazar entre los aviones de la forma más rápida posible. Si se considera que estas conexiones suelen realizarse en grandes aeropuertos, el tiempo necesario para que los pasajeros realicen el cambio puede llegar a ser considerable si estos se emplazan en puertas muy separadas entre sí. Como ejemplo se puede considerar que el aeropuerto de Madrid Barajas tiene 227 puertas repartidas en cuatro terminales que cubren una superficie de 940 000 m² [1], de esta forma, el tiempo que deberán andar los pasajeros dependerá de la localización relativa entre las aeronaves.

El crecimiento del tráfico

Un rápido vistazo a la cantidad de operaciones realizadas anualmente permite observar fácilmente que, hasta el comienzo de la pandemia, existía una clara tendencia al alza. El aumento de la competencia y la bajada de los precios de los vuelos propiciados por la aparición de las aerolíneas low cost permiten que volar en avión no sea un lujo ni un privilegio, por lo que la cantidad de clientes no deja de aumentar. A su vez, el aumento de popularidad de las

plataformas de compra on-line facilita también el incremento del número de operaciones de carga.

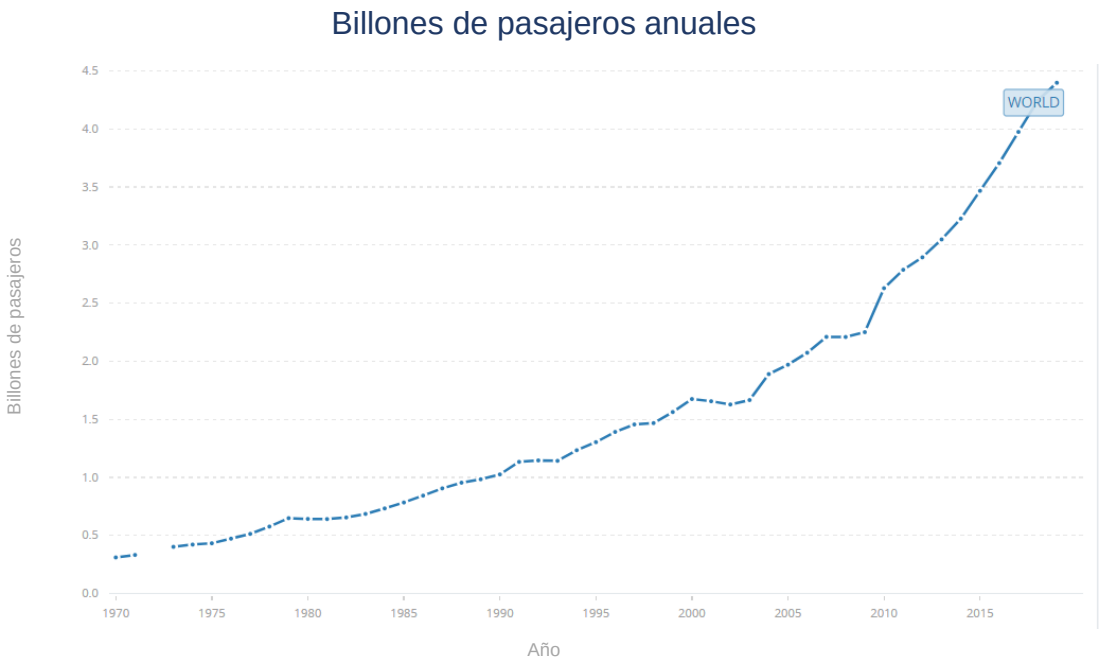


Imagen 1: Pasajeros anuales tráfico aéreo entre 1970 y 2019 [2]

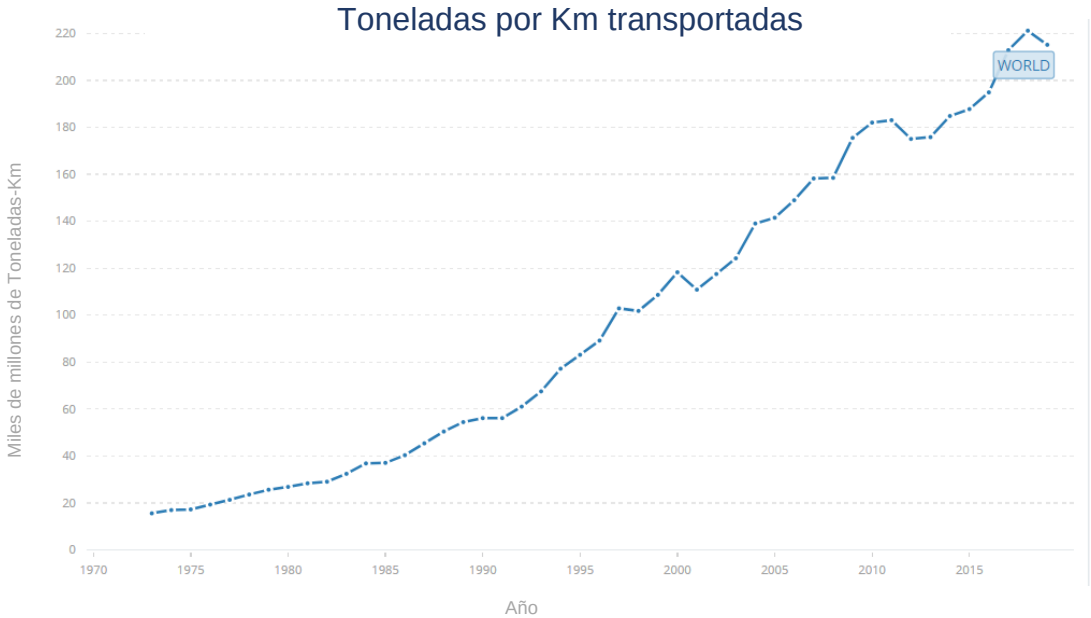


Imagen 2: Millones de toneladas de carga transportadas por Km entre 1970 y 2019 [3]

Viendo que la tendencia al alza es clara, se comprende perfectamente la importancia de este sector estratégico. Los operadores aeroportuarios son conscientes de este incremento de la demanda pero, dados los niveles de capacidad actuales, resulta muy complejo ampliarla. Por este motivo, resulta fundamental poder optimizar las instalaciones de las que ya se dispone antes de construir nuevas terminales. Cuanto más rápido se pueda atender a los aviones en su estancia en el aeropuerto, antes se quedará su puerta libre y se podrá albergar otro vuelo.

Retraso en las operaciones

Teniendo en cuenta que las operaciones aeroportuarias se comienzan a planear meses antes de su fecha, hay muchos factores que no se pueden prever y que, por tanto, son susceptibles de alterar los horarios. Los principales factores que pueden afectar a la duración de los vuelos son:

- Meteorología: aquí se deben considerar no solo los fenómenos adversos como tormentas que haya que esquivar, sino que en condiciones normales siempre existe viento que puede modificar la velocidad de la aeronave (generalmente TAS no equivale a Ground Speed). También en el entorno del aeropuerto puede haber malas condiciones y, por ello, que el aeropuerto tenga una capacidad menor de lo habitual. Es el caso de niebla, nieve o fuertes vientos cruzados.
- Congestión del espacio aéreo: en determinadas ocasiones se pueden generar retrasos debido a la capacidad de los espacios en un momento concreto. Esto puede ocurrir tanto en el espacio aéreo superior durante la fase en ruta, como en la aproximación a un determinado aeropuerto.
- Averías en aeronave o aeropuerto: si la aeronave tuviera alguna posible avería y tuviera que ser revisada por los mecánicos antes del despegue, probablemente se genere un retraso sobre el vuelo, incluso si al final no es necesario realizar ningún cambio. De la misma forma, se pueden producir incidentes en el propio aeropuerto, bien en la pista (aeronave fuera de la pista, incidente con los pasajeros, aterrizaje de emergencia…) o bien en la terminal (problemas con el sistema eléctrico, problemas relacionados con el personal, problemas de seguridad…).
- Retrasos causados por los pasajeros: más allá de los problemas que se pudieran ocasionar debido a las políticas de seguridad, hay que tener en cuenta que muchas veces las aerolíneas ofrecen billetes con transbordo. Esto significa que la compañía se responsabiliza de que el pasajero pueda llegar al destino y, por tanto, puede ocurrir que un vuelo tenga que esperar a los pasajeros de otro que llegue con retraso. De lo contrario, la aerolínea debería indemnizar al pasajero.

Son tantos los motivos que pueden retrasar un vuelo que es un fenómeno habitual. Analizando los datos de los aeropuertos españoles durante el año 2018 se observa que 22.4 millones de pasajeros se vieron afectados por retrasos o cancelaciones [4]. Analizando de forma separada los principales aeropuertos españoles (Tabla 1), se observa que no son situaciones esporádicas.

PORCENTAJE DE RETRASOS O CANCELACIONES EN 2018	
BARCELONA - EL PRAT (BCN)	25,2%
PALMA DE MALLORCA (PMI)	24,7%
IBIZA (IBZ)	22,5%
MENORCA (MAH)	20,5%
MÁLAGA - COSTA DEL SOL (AGP)	19,5%
ADOLFO SUÁREZ MADRID BARAJAS (MAD)	18,8%
ALICANTE - ELCHE (ALC)	18,3%
VALENCIA (VLC)	17,7%
SEVILLA (SVQ)	17,6%
TENERIFE SUR (TFS)	16,8%
BILBAO (BIO)	16,4%
TENERIFE NORTE (TFN)	14,6%
GRAN CANARIA (LPA)	14%
LANZAROTE (ACE)	13,6%
FUERTEVENTURA (FUE)	12,5%
LA PALMA (SPC)	11,9%

Tabla 1: Porcentaje de retrasos o cancelaciones en los principales aeropuertos españoles durante 2018 [4]

Tiempo de Turnaround

El llamado tiempo de Turnaround es el intervalo de tiempo que pasa entre la puesta de calzos en las ruedas de la aeronave (se considera que la operación ha finalizado) posterior al aterrizaje y la retirada de los mismos antes del despegue. Que este tiempo sea bajo es crucial tanto para la aerolínea, que pretende tener el máximo tiempo posible la aeronave volando, como para el gestor aeroportuario, que pretende albergar al máximo número de aeronaves posible. No obstante, se trata de un tiempo en el que se tienen que realizar una serie de operaciones complejas y que involucran a distintos participantes. Por una parte, están los agentes de handling, que son quienes realizan las operaciones sobre la aeronave. Algunas de sus funciones habituales son: descargar el equipaje, la carga y a los pasajeros (si no es una puerta con finger se tendrá que proporcionar una escalera), carga de combustible y agua, descarga de aguas sucias, limpieza de la cabina, recarga del cáterin y carga del nuevo equipaje y pasajeros. Por otra parte, durante el tiempo de Turnaround muchos pasajeros (o todos) abandonarán la aeronave y otros nuevos subirán.

Dado que las compañías quieren reducir este tiempo al mínimo, van a exigir que las puertas de embarque de los vuelos con transbordo estén cerca, facilitando así que los pasajeros y sus equipajes lleguen más deprisa. Poder reducir el tiempo de Turnaround puede suponer un ahorro importante, aunque resulta complejo determinar una cifra exacta ya que depende de múltiples factores. Lo que sí se conoce es que la reducción de este tiempo es una de las claves del éxito de las aerolíneas low cost como Ryanair o Wizzair que tienen tiempos de Turnaround de 25 y 30 minutos, la mitad del tiempo habitual de las aerolíneas tradicionales [5]. Así, se entiende la presión de las aerolíneas por tener puertas de embarque cercanas que permitan seguir reduciendo, si es posible, el tiempo de Turnaround. De la misma forma, el gestor aeroportuario

también está interesado en reducir este tiempo ya que, como se ha comentado, cuanto menos tiempo estén las aeronaves en la puerta, mayor capacidad podrá albergar el aeropuerto.

Capítulo 2
Formulación del problema

A lo largo del presente trabajo se va a exponer el conjunto de investigaciones y desarrollos realizados en el estudio sobre la optimización de las puertas de embarque en aeropuertos congestionados. Así pues, el estudio busca ofrecer la mejor solución a la asignación de puertas de embarque; es decir, una vez aterriza un avión concreto en un aeropuerto, determinar a qué puerta debe dirigirse. Para un pasajero que toma un avión puede parecer una decisión arbitraria realizada sin ningún criterio pero, como se verá a continuación, de la asignación realizada dependerá su comodidad o la posibilidad de sufrir un retraso, entre otros.

Así, el problema planteado considera que se conoce toda la información relativa a los vuelos y al aeropuerto. Por ello, los datos que se consideran como parámetros de entrada al problema serán:

- Distribución concreta de las puertas aeropuerto: esto se especificará con la distancia numérica entre las diferentes puertas de embarque disponibles. De igual forma, se podrá tener en cuenta la categoría de cada puerta y los vuelos que puede alojar tanto por tamaño como por características del vuelo (con control previo o no de pasaportes).
- Información relativa a los vuelos: esto supone que se incluye toda la información relativa a la planificación del uso de la pista del aeropuerto y que determina las horas de aterrizaje y despegue previstas para cada aeronave. De forma adicional, se estudiará la posibilidad de que dicha planificación se incumpla y que, por tanto, exista una determinada probabilidad de retraso o anticipo. Así, se llegará al punto en el que el problema considere las horas de aterrizaje y despegue, así como su variabilidad, distinta para cada vuelo.
- Información sobre los pasajeros que conforman cada uno de los vuelos: se requerirá también conocer la naturaleza de los vuelos en conexión (transbordo) con las relaciones de pasajeros que llegan en un vuelo y deben irse en otro que sale después.

De la misma forma, se deben considerar las restricciones existentes. De forma básica se considerará que una puerta puede albergar solo a un avión a la vez y que un avión debe estar asignado a una puerta obligatoriamente. De forma adicional existen otras restricciones marcadas por el carácter de los vuelos y por tanto algunos aviones no podrán ser asignados a algunas puertas.

Una vez se conoce toda esta información, será necesario conocer qué tipo de solución se quiere obtener. Para conocer los distintos tipos se deberá realizar un análisis de la bibliografía existente que permita observar cuáles son las ventajas e inconvenientes de cada asignación. Esto se concretará definiendo un problema de optimización en el que la asignación elegida definirá una variable que el problema maximizará o minimizará. Para resolver problemas complejos como los planteados se deberá estudiar el software existente para encontrar uno que permita diseñar los algoritmos de resolución de cada una de las asignaciones. El resultado

último de las asignaciones de cualquier tipo siempre será una relación de los vuelos que atender con la puerta a la que deberán dirigirse una vez hayan aterrizado.

Como se observará más adelante, aunque el problema resulte sencillo de comprender, para sistemas complejos donde el número de puertas y pasajeros es considerable, se encuentra que es un problema muy difícil de resolver puesto que la cantidad de soluciones posibles es enorme. Este detalle implica que estamos delante de lo que la teoría de la complejidad computacional denomina como un problema NP-hard. Por tanto, se trata de un problema para el que es casi imposible encontrar un algoritmo eficiente capaz de encontrar una solución óptima. De esta forma, cualquier algoritmo generado en realidad aportará soluciones aproximadas.

Como parte final del trabajo se abordará una comparación entre los diferentes tipos de asignaciones estudiadas que permita establecer las ventajas e inconvenientes de cada una de ellas. Para mayor claridad se deberá realizar un ejemplo numérico que, partiendo de unos mismos datos de entrada, muestre resultados de asignación distintos. Lo más interesante aquí será comparar los parámetros estudiados en cada una de las optimizaciones para ver cuánto empeoran en las otras asignaciones.

Capítulo 3
Estado del arte

Como ya se ha visto, del uso eficiente de los recursos aeroportuarios depende en gran medida el coste de las operaciones. Este problema para nada es nuevo ya que desde la liberalización del sector en los años 80, el tráfico ha crecido de forma considerable y se han comenzado a ver las primeras situaciones de congestión en los aeropuertos de muchas ciudades. Por este motivo, el problema de la asignación de puertas ("GAP" como se conoce por sus siglas en inglés) se ha estudiado en profundidad desde entonces mediante las herramientas matemáticas e informáticas disponibles en cada momento. La cantidad de factores a tener en cuenta y las diferentes partes interesadas provocan que el problema se pueda ver desde distintos puntos de vista. Esto es, en esencia, un problema de optimización, donde existe una función a maximizar o minimizar. A lo largo de esta sección se pretende exponer algunos ejemplos de estudios ya realizados que ejemplifican la disparidad de criterios al seleccionar el parámetro óptimo.

En primer lugar, se debe tener en cuenta que la función a optimizar puede ser distinta según los interesados. Así, se observa que hay sistemas centrados en optimizar la operación de las aerolíneas (minimización de la distancia que caminan los pasajeros o la distancia entre la pista de aterrizaje y la puerta) y sistemas centrados en optimizar la capacidad del aeropuerto (evitar conflictos o minimizar aviones en stand remotos). A lo largo de la siguiente clasificación en función de la naturaleza de los datos, se expondrán los distintos ejemplos de funciones a optimizar.

En segundo lugar, se puede definir una cuestión que cambia por completo el planteamiento del problema. De esta forma, atendiendo al tipo de variables utilizadas, se pueden definir dos estrategias distintas:

- Con datos deterministas, es decir, los datos se consideran invariantes y fijos. Las variables deterministas plantean el problema de la forma más simple, ya que se asume que todas las restricciones siempre son ciertas. Esto incluye asumir que las horas de llegada y salida de los aviones siempre se van a corresponder con la realidad. El principal problema que presenta esta metodología es que para aeropuertos concurridos las soluciones carecen de robustez. Esto significa que la más mínima desviación en algún parámetro puede hacer que sea imposible llevar a cabo la solución que el algoritmo plantea.
- Con datos probabilísticos, es decir, los datos de entrada se ofrecen como una función de distribución de probabilidad de la presencia de la aeronave en la puerta. Lo más habitual es que se consideren distribuciones gaussianas caracterizadas por promedio y varianza. Este tipo de problemas permitirá añadir la robustez necesaria a las soluciones, aunque todo ello será a costa de aumentar la complejidad del sistema y la cantidad de datos necesarios para abordarlo.

Soluciones deterministas

Como ya se ha visto, la solución determinista del problema considera que las horas de llegada y salida de los aviones son invariantes y conocidas de antemano. De esta forma el sistema se concibe como bloques que deben ser encajados, uno detrás de otro, en alguna de las puertas. De forma adicional, se definen restricciones relativas al tamaño del avión o la categoría del vuelo. Podemos ver un diagrama que muestra el problema de forma simplificada en la Ilustración 1. Aquí se ha modelado una franja temporal para dos puertas de cada uno de los 4 tipos básicos y una serie de aviones que están en ellas desde una hora de inicio hasta una hora de salida (slots).

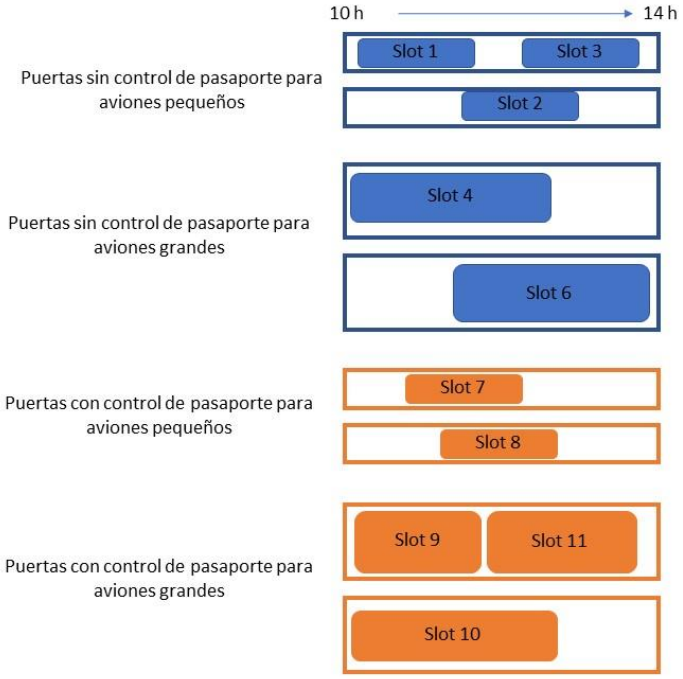


Ilustración 1: Diagrama de bloques de una asignación determinista

No obstante, si se considera un escenario real con un número elevado de puertas y slots, la asignación puede tener distintas soluciones. De entre todas ellas, el sistema de asignación buscará la que optimice algún criterio en concreto. A lo largo del tiempo, se han explorado distintas posibilidades y cada estudio define unas restricciones a aplicar y la función que se pretende maximizar. Los siguientes apartados exponen algunos ejemplos:

Soluciones deterministas que minimizan la distancia recorrida por los pasajeros

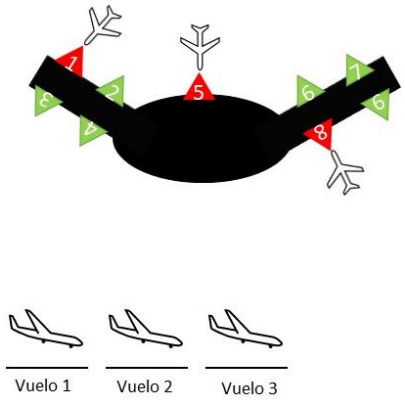
Las primeras soluciones que se dieron al problema perseguían el objetivo de minimizar la distancia que los pasajeros deben recorrer por la terminal, ya sea para coger otro avión (pasajeros en tránsito), para recoger el equipaje o para llegar desde el control de seguridad. En este caso, tal y como hace el ejemplo de la Ilustración 1, se consideran problemas distintos en

función de las restricciones y cada combinación de tamaño y tipología de puerta se resuelve de forma independiente.

De todas las soluciones estudiadas, esta es la más simple. Sobra recordar que la capacidad computacional para resolverlos se ha incrementado sobremedida en los últimos años, por lo que se entiende que en el momento de publicación de los primeros estudios sencillamente no era posible pensar en las soluciones más modernas. Hay que destacar que la solución expuesta a continuación fue publicada en 1990.

Pese a ser una solución sencilla, minimizar la distancia que andan los pasajeros puede resultar muy beneficioso para los aeropuertos más concurridos o para los que tienen mucho tráfico en tránsito, que habitualmente suelen ser los más complejos de organizar. Del tiempo que tarden los pasajeros en moverse por la terminal dependerán muchos de los factores importantes como el tiempo mínimo de "Turn-Around" (el avión saliente debe esperar a que lleguen los nuevos pasajeros), el confort de los pasajeros o la utilización de la terminal (flujos de pasajeros) y, con ello, las posibles colas generadas.

Como punto de partida para comprender cómo se resuelve el problema se va a exponer la solución propuesta en el libro *Airline operations and scheduling* [6] . El problema parte de una serie de puertas que están disponibles en un determinado instante de tiempo (las otras ya están ocupadas) y una serie de vuelos que van a llegar próximamente con pasajeros con destino a distintas puertas. De la misma forma, el planteamiento asume que cualquiera de los vuelos se puede alojar en cualquiera de las puertas vacías (no existen restricciones de tamaño o categoría). Toda esta información se codifica mediante las matrices de flujo de pasajeros y de distancia entre puertas, como se representa en la Ilustración 2. En ella vemos cómo hay algunas puertas de la terminal que ya están ocupadas con vuelos que van a salir próximamente y una serie de vuelos que van a llegar y tienen pasajeros que deben coger otro vuelo de los que están a punto de salir.



Matriz distancias entre puertas (unidades de longitud)					
Origen/destino	Puerta 1	Puerta 2	Puerta 3	...	Puerta k
Puerta 1		$dP_{1,2}$	$dP_{1,3}$	...	$dP_{1,k}$
Puerta 2	$dP_{2,1}$		$dP_{2,3}$	...	$dP_{2,k}$
Puerta 3	$dP_{3,1}$	$dP_{3,2}$		...	$dP_{3,k}$
⋮	⋮	⋮	⋮		⋮
Puerta j	$dP_{j,1}$	$dP_{j,2}$	$dP_{j,3}$	...	$dP_{j,k}$

Matriz flujo de pasajeros (N.º pasajeros)					
Vuelo/ Destino	Puerta de destino 1	Puerta de destino 2	Puerta de destino 3	...	Puerta de destino k
Vuelo 1	$P_{1,1}$	$P_{1,2}$	$P_{1,3}$	...	$P_{1,k}$
Vuelo 2	$P_{2,1}$	$P_{2,2}$	$P_{2,3}$	...	$P_{2,k}$
Vuelo 3	$P_{3,1}$	$P_{3,2}$	$P_{3,3}$	...	$P_{3,k}$
⋮	⋮	⋮	⋮		⋮
Vuelo i	$P_{i,1}$	$P_{i,2}$	$P_{i,3}$	...	$P_{i,k}$

Ilustración 2: Datos iniciales para la asignación de mínima distancia

Con estos datos, se puede calcular la distancia total que deberán andar los pasajeros como $Distancia\ total = \sum número\ de\ pasajeros * distancia$. Esta es la función que el problema busca minimizar.

Pese a ser un modelo que permite obtener la distancia mínima de forma sencilla, se observa que no se cumplen muchos de los objetivos del problema. De forma general se encuentran las siguientes desventajas que alejan al sistema de ser realmente óptimo:

1. Se consideran todas las puertas iguales. Esto eliminaría, por ejemplo, la posibilidad de introducir un avión de tamaño pequeño en una puerta preparada para aviones de tamaño grande que estuviera disponible y resultara conveniente.
2. Los pasajeros en tránsito están asignados a una puerta y no a otro vuelo. De esta forma, ninguno de los pasajeros podría subirse a otro vuelo que llegue a la vez.
3. Está planteado para grupos de vuelos que vienen a ocupar los huecos libres. Dichos huecos se han establecido bajo otros criterios. Así, la solución que da el problema puede ser buena para alojar a un primer grupo, pero resultar perjudicial para los siguientes procesos, ya que deja los huecos sin contemplar las situaciones futuras.
4. Cada puerta puede alojar solo un vuelo, sin tener en cuenta que, si el periodo de tiempo es muy largo, el avión podría irse y dejar vacío el hueco.

Con todo ello se observa que la solución no ofrecerá buenos resultados para una planificación completa.

Vistas las limitaciones del modelo, resulta interesante explorar otras opciones. Una de las más completas está contenida en [7], donde se modela una variable de decisión para el vuelo de origen y su puerta y otra para el vuelo de destino y su puerta. Gracias a la inclusión de esta segunda variable, se permite introducir los parámetros de entrada como pasajeros del vuelo X que quieren ir al vuelo Y. Además, se introducen otros parámetros como las puertas en las que se podrá albergar cada avión o las horas de llegada y salida que permiten asignar varios vuelos a una misma puerta siempre que no compartan horario. Gracias a estos avances, se encuentra un sistema de asignación muy eficaz que ofrece una asignación óptima para todo un día y que tiene en cuenta muchas de las limitaciones que se encuentran en el mundo real. No obstante, el modelado que se muestra en [7] se resuelve por métodos de iteración, ya que del modo que está definido no se puede resolver mediante programación lineal. De este modo, antes de estudiar el rendimiento de esta asignación, habrá que generar un modelo apropiado que pueda ser resuelto mediante programación lineal.

Otras soluciones deterministas

Una vez vista la organización general del modelo, hay que resaltar que existen otros modelos que, partiendo de estos, añaden ligeras variaciones o incluso plantean optimizar otras variables. Algunos ejemplos que existen son: minimizar los vuelos sin puerta asignada (y por tanto deben utilizar stands remotos), los tiempos que deberán estar los aviones en la puerta o los que computan también las distancias recorridas por los pasajeros con origen o destino en el propio aeropuerto (y no solo los pasajeros en tránsito). Esta disparidad se debe a que, al menos durante los años de publicación, no era posible llegar a una solución óptima que

considerara todas las restricciones a la vez utilizando los métodos tradicionales de programación lineal. Con esto se concluye que la mayoría de los estudios solía seguir una metodología mixta en la se incorporaban técnicas heurísticas o de simulación para llegar a soluciones de optimización particularizadas a un aeropuerto o terminal.

Con el paso del tiempo, las limitaciones computaciones desparecían y se propuso ampliar las metodologías anteriores. Aparecen entonces otras propuestas que amplían las restricciones y permiten asignar vuelos de categoría menor a puertas más grandes si esta está vacía y esto resulta óptimo. De esta forma, el sistema se evaluaba como un solo conjunto y no como distintos grupos de puertas independientes. Otras restricciones que se han estudiado [6] en estos sistemas ha sido introducir en la ecuación a minimizar la distancia recorrida por los equipajes, ya que desde el punto de vista de las operaciones de handling existen situaciones en las que este parámetro puede resultar limitante debido a la distribución interna del aeropuerto. De nada sirve que los pasajeros lleguen muy rápido a su destino si van a tener que esperar porque su equipaje no ha llegado. De esta forma, el sistema busca optimizar la suma de las distancias recorridas por los pasajeros y las recorridas por las maletas, que además pueden tener prioridades distintas (multiplicando cada distancia por un parámetro distinto).

Otra aproximación similar basada en la optimización mediante programación lineal es la que se encuentra en la propuesta de [8] en la que se propone cambiar distancia por tiempo, asumiendo una velocidad media de los pasajeros. En este artículo, los autores asumen que lo más importante para el pasajero es la percepción psicológica y, por tanto, aunque el tiempo transcurrido sea el mismo, no se percibe de igual forma si el pasajero está andando o esperando para poder recoger su equipaje. Con todo esto, el criterio a minimizar es el "tiempo de retraso de puerta" calculado de forma independiente para cada puerta como la diferencia entre el tiempo que tarda el pasajero en llegar a la zona de recogida de equipajes y el tiempo que tarda en llegar el equipaje. Cabe destacar que este sistema solo tiene en cuenta a los pasajeros que abandonan el aeropuerto ya que los que están en tránsito generalmente no recogen el equipaje. Este caso asume, por tanto, que el traslado de equipajes entre aeronaves siempre es más rápido que el de pasajeros.

La preocupación por la robustez

Con el paso del tiempo, los que estudiaban el problema se dieron cuenta de que las asignaciones que estaban realizando, pese a ser óptimas, sufrían grandes problemas cuando alguno de los vuelos sufría cualquier tipo de retraso o avance. La capacidad del sistema a soportar variaciones en los parámetros de entrada es, en esencia, la robustez del mismo. Para aumentar la robustez del sistema, se proponen referencias bibliográficas [9] en las que el objetivo de la solución es maximizar el tiempo que la puerta está desocupada entre vuelos ("idle time"), garantizando así el máximo tiempo entre vuelos consecutivos para obtener una asignación más robusta. No obstante, en las publicaciones anteriores a esta se encontraba con una limitación de 20 puertas y 80 aviones en una misma ventana de tiempo mediante la resolución con programación lineal (año 2002). De esta forma, esta publicación [9] presenta una solución para, aproximadamente, optimizar el tiempo en el que la puerta está desocupada. Dicha aproximación supone que la función optimizada no es directamente el tiempo de

desocupación; no obstante, la solución se aproxima mucho y permite un cálculo a mayor escala. Durante esta asignación, los vuelos se asignan a una puerta cualquiera (considerando el número de puertas de cada tipo del que se disponen), por lo que durante una segunda fase se determina qué puerta concreta seguirá cada plan. En el artículo [9] , se sigue para esta segunda fase el criterio de mínima distancia recorrida por los pasajeros en la terminal. Después de esto se podría, de forma manual, realizar algún cambio como, por ejemplo, asignar un avión pequeño a una puerta inicialmente reservada para aviones grandes ya que su algoritmo no lo permite.

Resumen publicaciones deterministas			
Autor	Objetivo	Detalles	Problemas
Bazargan, M. [6]	Mínima distancia recorrida por los pasajeros de transbordo	Asumiendo que ya existen unos vuelos en el aeropuerto se asignan puertas a los que van a llegar próximamente. Los pasajeros se asignan a una puerta	- Todas las puertas iguales - Los pasajeros hacen tránsito con aviones que hayan llegado antes que el suyo. No podrían esperar a uno que aterrice a la vez o después - Los huecos que se dejan pueden ser perjudiciales para los siguientes vuelos. Se perjudica a los grupos de vuelos que llegan más tarde - No existe la posibilidad de que vuelos con distintos horarios compartan puerta - No existe robustez
Thanyan AL-Sultan, A. [7] Ding, H., Lim, A., Rodrigues, B. & Zhu, Y. [10]	Mínima distancia recorrida por todos los pasajeros. Evita que haya vuelos asignados a puertas remotas	Asigna hora de llegada y salida de cada vuelo. Si llega cuando el primero se ha ido se permite compartir puerta. Los pasajeros tienen destino otro vuelo y no una puerta	- Se resuelve con algoritmo propio. Los métodos generales de programación lineal no hallan respuesta - No existe robustez. El hueco puede ser de pocos minutos
Gürsoy, M., & Akyildlz Alçura, G. [8]	Mínimo tiempo de espera de los pasajeros (*Gate delay Time*)	Asume que lo más importante para los pasajeros no es cuánto anden, sino el tiempo que esperan quietos. Considera el tiempo que tardan los pasajeros en llegar y el tiempo que tardan las maletas en salir	- No tiene en cuenta a los pasajeros en tránsito - No se considera la cantidad de pasajeros en cada vuelo - No existe robustez
Utrecht University [9]	Máximo tiempo de desocupación de la puerta entre vuelos (*Idle time*). Después, se busca la mínima distancia de pasajeros	La asignación se hace en dos fases: primero asigna aviones a puertas genéricas, después determina qué puerta física seguirá cada horario	- No se puede computar directamente el tiempo entre vuelos - Aunque es robusto no contempla que cada vuelo tiene distinta probabilidad de retrasarse

Tabla 2: Resumen de las publicaciones deterministas existentes

Soluciones probabilísticas

Siguiendo esta tendencia a buscar sistemas robustos y resistentes a cambios, se ha llegado a sistemas modernos como el presentado en [11], que va más allá del método de maximizar los huecos (*"idle time"*) mencionado anteriormente ya que tiene en cuenta que la probabilidad de retraso de cada vuelo puede ser distinta. En este sentido, este método cambia por completo el planteamiento del problema pasando de ser determinista (datos cerrados) a ser estocástico (considerando la varianza de los datos). El método que se utiliza es eliminar la restricción de tener un solo avión por puerta y pasa a considerarse la probabilidad de que exista concurrencia en las puertas: multiplicación de las probabilidades de ocupación de cada uno de los vuelos. Así, la publicación [11] parte de un nivel de concurrencia máximo determinado por el usuario que no se podrá superar en ninguna puerta en ningún instante. Una vez se ha comprobado esto, se realiza la asignación buscando optimizar un factor coste no especificado en el artículo pero que podría hacer referencia a la distancia caminada por los pasajeros, por ejemplo.

De esta forma, en función del nivel de robustez prestablecido se podrá decidir si es conveniente aceptar más vuelos o no. La principal ventaja de este sistema es que, para estudiar la probabilidad de concurrencia, se tendrá en cuenta el posible retraso de la aeronave en función de la distancia, destino, aerolínea, etc. Como es evidente, este tipo de sistemas puede ofrecer soluciones muy buenas, aunque para ello se deberá considerar una gran cantidad de datos históricos y hacer un procesamiento correcto y computacionalmente costoso. Como conclusión de la investigación, se obtiene que al utilizar este método, se deberá elegir los parámetros en base a un compromiso entre robustez y eficiencia del sistema.

Otros estudios buscan disminuir las posibilidades de lo que llaman *bloqueo* (concurrencia de dos vuelos en un mismo instante en una puerta) como se muestra en [12] donde se explican distintas formas de considerar el perjuicio de los bloqueos. Así, se observan las diferentes medidas de la robustez:

- Probabilidad de bloqueos: se pretende aquí minimizar la probabilidad de que dos vuelos deban estar simultáneamente en la puerta.
- Tiempo total de bloqueo: se pretende aquí minimizar los bloqueos más largos frente a los cortos. Este método completa el anterior para el que no se tenía en cuenta el tiempo durante el que existía el bloqueo.
- Minutos de bloqueo sobre los pasajeros: se pretende aquí minimizar el tiempo extra que los pasajeros en tránsito tardarán en llegar a su puerta de destino, ya que son los más perjudicados por los bloqueos.
- Tiempo de bloqueo máximo: priorizando así que los bloqueos sean cortos frente a que haya pocos bloqueos.

Una vez explicado esto, el artículo ofrece algunos detalles sobre la modelización matemática (aunque de forma general con una función coste) y las restricciones impuestas para casos en los que cualquier aeronave puede usar cualquier puerta y para casos en los que se contemplan limitaciones en la asignación.

Pese a haber encontrado soluciones deseables desde el punto de vista del operador, el problema no se puede considerar como resuelto, puesto que para cada uno de los agentes interesados los parámetros importantes son distintos. En este sentido, otros estudios como el presentado en [13] admiten que, desde el punto de vista de las aerolíneas y de algunos pasajeros, resulta mucho más práctico tener una misma puerta para los vuelos periódicos. Si se añade esta restricción en el algoritmo, el problema pasa a tener un tiempo de estudio mucho más largo ya que la asignación de un día puede tener consecuencias sobre la asignación del siguiente. No obstante, la publicación tan solo plantea la posibilidad de añadirlo, aunque no lo lleva a cabo. Por este motivo, no se pueden extraer conclusiones claras acerca de sus posibles perjuicios sobre la robustez u optimización global de cada uno de los días.

Sistemas expertos

Pese a la considerable investigación realizada a lo largo de los años, todavía no se ha encontrado ningún método capaz de resolver la asignación de forma completamente óptima. No obstante, la asignación de puertas hoy en día es muy eficiente ya que los aeropuertos utilizan software específico desarrollado expresamente para esta tarea. Así, utilizan técnicas de inteligencia artificial de toma de decisiones que van más allá de la optimización de una cierta variable en concreto. Son lo que se denominan sistemas expertos. Estos, permiten tomar decisiones como lo haría un humano experto (de ahí su nombre) aunque para ello se necesita una base de datos de conocimiento muy grande que permita que el sistema "aprenda" lo suficiente como para realizar una buena asignación.

Aunque el software más utilizado en España es el *Sally* desarrollado por SITA, existen otros desarrollados por empresas como Indra, Gentrack, Siemens o Amadeus entre otros. Dado que se trata de soluciones comerciales resulta complejo saber cómo funcionan sus algoritmos o en qué principios se basan por lo que no es posible contemplar esta posibilidad en el desarrollo del sistema propio.

Resumen publicaciones probabilistas			
Autor	Objetivo	Detalles	Problemas
Schaijk, O. R., & Visser, H. G. [11]	Función coste no especificada	El usuario determina un nivel máximo de probabilidad de concurrencia y una de las restricciones asegura que no se supere en ningún caso.	- La solución dependerá de un parámetro introducido por el usuario que se deberá establecer de forma experimental. - Es difícil determinar la relación robustez-influencia en el coste - No se especifica cómo determinar el coste de cada solución
Castaing, J., Mukherjee, I., Cohn, A., Hurwitz, L., Nguyen, A. & Müller, J. J. [12]	Se proponen distintos tipos: - Probabilidad de bloqueos - Tiempo total de bloqueo - Minutos de bloqueo sobre los pasajeros en tránsito - Tiempo de bloqueo máximo	La publicación no proporciona información acerca del modelado de cada uno de los tipos, utilizando un parámetro coste genérico para definir el modelo.	- No se tienen en cuenta las distancias de los pasajeros ni la comodidad de los tránsitos. Ni siquiera en el caso del tiempo de bloqueo a los pasajeros se tiene en cuenta ya que solo se estudia el tiempo extra en llegar a la puerta, no el del traslado por la terminal.
Pesch, E., Dorndorf, U., & Jaehn, F. [13]	Optimizar varios parámetros ponderados (función coste)	Se propone realizar el estudio de forma semanal para asignar vuelos periódicos a mismas puertas	- No se especifica cómo modelar el sistema - Este sistema penaliza a los vuelos no periódicos y les asignará las peores puertas

Tabla 3: Resumen de las publicaciones probabilistas existentes

Capítulo 4
Métodos de resolución

Los problemas de optimización no son más que una técnica matemática que permite tomar decisiones concretas y objetivas en problemas asociados a condiciones reales. Así, el primer paso consistirá siempre en saber transformar el problema del lenguaje normal al lenguaje matemático y así, mediante distintas operaciones, obtener el resultado deseado. A esta transformación se le llama construir el modelo del problema. De una forma más concreta se han definido las distintas fases a seguir durante el modelado [14]:

1- Identificar las variables del problema: Determinar qué es lo que se quiere que el problema determine. Existe un gran número de posibles soluciones a estas variables, aunque no todas serán válidas.
2- Determinar qué decisiones resultan admisibles. Se debe especificar mediante lenguaje matemático qué condiciones debe cumplir la asignación de las variables para que la solución sea válida. Generalmente se incluyen aquí sumatorios o productos de las variables como parte de desigualdades. El resultado de esta fase es el establecimiento de las restricciones del problema.
3- Determinar el coste/beneficio de cada posible solución. Para ello se determina una función objetivo que representa lo que se quiere que sea máximo o mínimo. Un ejemplo muy típico del dominio de esta función suele ser la minimización del coste económico o la maximización de los beneficios.

Las técnicas utilizadas suelen tener carácter iterativo y ser distintas en función del tipo de variables y datos del problema. Atendiendo al tipo de funciones utilizadas en las restricciones y en la variable a optimizar, se encuentran:

- Problemas de programación lineal: las funciones son lineales
- Problemas de programación no lineal: existe alguna (o varias) función no lineal. Aunque los problemas lineales cubren gran parte de las aplicaciones, en ocasiones es necesario incluir alguna función no lineal. Son comunes en esta categoría los problemas relacionados con geometría (como la optimización del volumen o la superficie), mecánica o electricidad.

Atendiendo al tipo de variables, se definen los siguientes tipos de problemas:

- Problemas con variables reales: Las variables pueden tomar cualquier valor real, por lo que se asume continuidad en las funciones. Este tipo se suele denominar programación lineal o no lineal según el caso, asumiendo que, si no se especifica, es porque las variables son reales.
- Programación con variables enteras: se utilizan solo variables enteras o binarias (pueden tomar valor 0 o 1) y, por tanto, no existe continuidad entre las soluciones. Es posible que existan variables tanto enteras como reales, por lo que se suele hablar de programación entera-mixta. Dentro de esta categoría, se encuentra también la posibilidad de utilizar lo que se llaman métodos relajados. Estos métodos se basan en

la *relajación* de algunas restricciones en las primeras iteraciones del algoritmo, para más adelante ir añadiendo nuevas restricciones que suplan la *relajación* inicial.

Una vez se ha completado el modelado y la definición del problema, se debe seleccionar algún algoritmo de resolución que busque la solución óptima. A la hora de resolver este tipo de problemas de optimización, existen diferentes estrategias y métodos. Se pueden clasificar en las siguientes categorías [15]:

- Métodos exactos: son aquellos que exploran todas las posibles soluciones del sistema, y de todas ellas eligen la que realmente es la más óptima. Como se ha mencionado, el problema es un NP-hard y por tanto explorar todas las soluciones no es una opción factible para el problema de optimización de puertas a gran escala ya que el tiempo de ejecución sería inasumible.
- Métodos heurísticos: son métodos simples e intuitivos que permiten encontrar una solución buena en un tiempo de ejecución razonable. Así, en vez de explorar todas las soluciones posibles, se estudian algunas de forma aleatoria o mediante inteligencia artificial básica.
- Métodos metaheurísticos: estos métodos son la evolución de los heurísticos ya que aplican algoritmos de búsqueda con un nivel de inteligencia artificial mayor y por tanto permiten obtener mejores resultados.

La herramienta de programación GAMS

Para resolver los problemas de programación lineal existe multitud de software de computación matemática disponible capaz de resolver los modelos más sencillos. No obstante, cuando se trata de modelos con una cantidad de ecuaciones y variables considerable, puede resultar muy complejo tener que introducir cada una de ellas de forma individual. Para problemas como los surgidos en el presente estudio se requiere de una herramienta capaz de generar automáticamente todas las variables y ecuaciones una vez se indique la cantidad deseada. De entre los programas software disponibles se ha escogido la herramienta GAMS, que destaca por su facilidad de uso en la programación, ya que el lenguaje que utiliza se asemeja mucho al matemático. No obstante, la utilización de un software nuevo siempre requiere un pequeño aprendizaje previo. Para ello se han utilizado algunas partes del manual proporcionado por el desarrollador [16].

Las siglas de GAMS provienen del nombre General Algebraic Modeling System ya que es un software desarrollado expresamente para "*el modelado, análisis y resolución de problemas de optimización*" [14] de distintos tipos. Como ya se ha visto, su principal ventaja es que su lenguaje se asemeja mucho a la descripción matemática típica de los problemas de optimización y, por tanto, lectores no familiarizados con el mismo podrán comprender su programación. Además, se pueden destacar otras ventajas como la capacidad para pasar de un problema de pequeña dimensión a uno mucho más complejo sin variar gran parte del código. Estos casos solo requerirán de la inclusión de los nuevos datos y variables, pero el código asociado a las definiciones y ecuaciones será prácticamente idéntico. Igualmente, el software permite un uso muy eficiente de los índices, por lo que se pueden agrupar restricciones y ecuaciones fácilmente. Además, es una herramienta muy versátil ya que permite que el usuario

elija el tipo de solución buscada pudiendo elegir entre una notable cantidad de algoritmos de resolución (*solver*) que se han desarrollado expresamente para cada uno de los distintos tipos de problemas. De igual forma se permite ejecutar el código en GAMS mediante algoritmos de otros programas como CPLEX o MATLAB entre otros.

Dado que la cantidad de funciones del programa es importante, se van a describir brevemente solo aquellas que se utilizarán en el modelado del problema de asignación de puertas. Lo primero que se debe conocer es la estructura típica del código y para ello resulta útil observar el ejemplo del transporte proporcionado por el desarrollador [16]. Este ejemplo considera que existen una serie de productores (*canning plants*) y consumidores (*markets*) separados por una distancia concreta que producen y consumen unas cantidades concretas de producto. Transportar el producto de los productores a los consumidores tiene un coste proporcional a la distancia. El problema consiste por tanto en encontrar la cantidad de producto que se debe transportar desde cada productor para satisfacer la demanda con el mínimo coste de transporte. El lenguaje se asemeja tanto al lenguaje matemático que una vez entendido el contexto resulta sencillo interpretar el código:

```
Set
   i 'canning plants' / seattle,  san-diego /
   j 'markets'        / new-york, chicago, topeka /;

Parameter
   a(i) 'capacity of plant i in cases'
        / seattle    350
          san-diego  600 /

   b(j) 'demand at market j in cases'
        / new-york   325
          chicago    300
          topeka     275 /;

Table d(i,j) 'distance in thousands of miles'
                          new-york  chicago  topeka
   seattle                 2.5       1.7       1.8
   san-diego               2.5       1.8       1.4;

Scalar f 'freight in dollars per case per thousand miles' / 90 /;

Parameter c(i,j) 'transport cost in thousands of dollars per case';
c(i,j) = f*d(i,j)/1000;

Variable
   x(i,j) 'shipment quantities in cases'
   z      'total transportation costs in thousands of dollars';

Positive Variable x;

Equation
   cost      'define objective function'
   supply(i) 'observe supply limit at plant i'
```

```
   demand(j) 'satisfy demand at market j';

cost..      z =e= sum((i,j), c(i,j)*x(i,j));

supply(i).. sum(j, x(i,j)) =l= a(i);

demand(j).. sum(i, x(i,j)) =g= b(j);

Model transport / all /;

solve transport using lp minimizing z;

display x.l, x.m;
```

Aquí se observan los distintos tipos de elementos según su función:

- Introducción de "actores del problema" y variables: el comando **Set** permite introducir los índices que permitirán recorrer los vectores del problema y asignar un nombre a cada uno de ellos y el comando **Variable** permite especificar las variables del problema. Aquí se encuentra una de las grandes virtudes de GAMS: la posibilidad de introducir elementos ordenados dentro de un set de forma sencilla. Así, si se quieren definir elementos como P1, P2, P3, P4… hasta PN se puede simplificar el código introduciendo el código /P1*PN/. Por otro lado, las variables serán lo que debe resolver el problema y, por tanto, la solución buscada. El programa también permite introducir el tipo de variable en cada caso pudiendo especificar si se trata de números enteros, positivos o binarios, entre otros, así como unos límites superiores e inferiores si se desea.
- Introducción de datos: los comandos **Parameter**, **Table** y **Scalar** permiten introducir los parámetros de entrada diferenciando las propiedades de cada uno de los índices definidos. Los tres comandos funcionan de forma similar, aunque se diferencian en la dimensión de los datos introducidos pudiendo ser vectores, matrices o escalares. Como se observa, se debe especificar un nombre del parámetro (con el que se introducirá en las ecuaciones) y los índices sobre los que está definido. Además de poderse introducir un valor numérico para cada índice (si sobre algún índice no se declara, se asume cero), se pueden introducir relaciones matemáticas entre otros parámetros introducidos anteriormente.
- Introducción de restricciones y relaciones entre variables: el comando **Equation** permite introducir qué condiciones deben cumplir las variables para que la solución sea aceptable y las relaciones entre las distintas variables. Como se observa en el ejemplo, primero se debe definir la ecuación con un nombre y los índices sobre los que se generará (de forma interna habrá una ecuación por cada combinación de índices) y a continuación se expresa su relación matemática. En este apartado quizá resulte extraña la forma de indicar la igualdad o desigualdad. La siguiente tabla relaciona los símbolos matemáticos con el código equivalente:

Símbolo matemático	Significado	Código en GAMS
=	Igualdad	=e=
≤	Menor o igual que	=l=
≥	Mayor o igual que	=g=

Tabla 4: Relación de símbolos matemáticos con su código utilizado en las ecuaciones en GAMS

- Especificación del tipo de problema, algoritmo de resolución y variables a mostrar: estas funciones permiten que el usuario indique cómo se debe resolver el problema y cuál es el objetivo buscado. Esto se realiza mediante los comandos **Model**, **Solve** y **Display** en el lenguaje del programa. El comando Model permite elegir que solo algunas de las restricciones introducidas formen parte del modelo (se podría resolver en un mismo fichero distintos modelos). No obstante, los problemas ejecutados en este estudio siempre consideran que el modelo contiene a todas las ecuaciones al igual que el ejemplo. El comando Solve permite elegir para uno de los modelos especificados qué variable se quiere optimizar y cómo (minimizar o maximizar) y qué tipo de problema y variables se encuentran en él. Los tipos de problema se corresponden con distintas agrupaciones de tipos de variables y funciones explicadas anteriormente. Aunque existen diez tipos distintos, el problema de la asignación de puertas se ha resuelto en todos los casos como un problema de Programación no lineal entera-mixta no relajada, lo cual equivale al código using minlp. Por último, el comando Display permite elegir qué variables se quiere que el programa muestre en el fichero de resultado. En programas sencillos como el del ejemplo se muestra solamente el valor de las variables solución, pero podría resultar útil mostrar también el valor del parámetro a optimizar o algún parámetro que se haya generado internamente mediante ecuaciones. Como se observa en el ejemplo, para visualizar la variable x se introducen dos elementos: x.l y x.m. El primero hace que se muestre la variable x en su estado final (para la solución óptima) y el segundo muestra el valor marginal de la variable asociado a la respuesta final. De forma adicional se pueden incluir otros subíndices que muestran el rango superior o inferior de las variables entre otras funciones, aunque para el problema estudiado no resulta de utilidad.

Pese a que estos son los elementos básicos del programa, las posibilidades son enormes ya que también se incluye la posibilidad de generar bucles, asignaciones complejas, incluir funciones matemáticas, expresiones condicionales, booleanos… A continuación, se van a exponer algunas de estas funciones que se han utilizado en el modelado de la asignación de puertas:

- Operadores de sumatorio que facilitan la comprensión y reducen la longitud de las expresiones. Su inclusión en las ecuaciones de GAMS es muy sencilla ya que solo se debe indicar el sumando y los índices para los que se quiere sumar. Así, la expresión **sum**(j, x(i,j)) refleja el sumatorio de los valores de la variable x para todos los elementos del índice j.
- Inclusión de librerías de funciones matemáticas. GAMS permite generar ficheros con funciones propias para incluirlas en las ecuaciones del programa. De igual forma, hay algunas funciones cuyo uso es frecuente que se encuentran de forma nativa en el programa, como las ecuaciones trigonométricas o las ecuaciones de densidad de

probabilidad de las principales distribuciones (normal, binomial, beta, cauchy…). Estas últimas resultan de especial utilidad para los problemas estocásticos como el que se presenta en la segunda parte del estudio de puertas de embarque. Para incluir estas funciones en un problema se debe primero incluir la librería ($FuncLibIn stolib stodclib) y luego declarar la función y el nombre con el que se le va a llamar (Function pdfNormal /stolib.pdfNormal/;).

- Computación de la posición de un elemento en un conjunto ordenado. Si los elementos definidos en los diferentes índices se corresponden con una asignación numérica ordenada, se podrá obtener su posición relativa mediante el comando ord(i).
- Introducción de expresiones condicionales. Aunque son muchas las utilidades del operador condicional $, en el modelado de la asignación de puertas solamente se usa un caso concreto: especificar que solo se quieren algunos elementos de un sumatorio. Para ello se utiliza un código con la siguiente estructura **sum**(j$(ord(j) > ord(i)), x(i,j)). En este ejemplo se indica que se haga el sumatorio de x(i,j) tan solo para los valores de j en los que se cumpla que el orden del índice j sea estrictamente mayor que el orden del índice i.

El algoritmo Antigone

A la hora de resolver los problemas, se pueden utilizar distintos algoritmos que pueden proporcionar una respuesta más optimizada o rápida en función del tipo de problema planteado. Dado que los problemas que se van a tratar en este estudio son de tipo no lineal cabe recordar que existen dos tipos de soluciones [16]: locales o globales. Una solución local implica que el punto de la gráfica es mayor o menor (según sea el objetivo) que los puntos a su alrededor, pero es posible que exista otro punto alejado todavía más pequeño o grande. Dado que el algoritmo por defecto del programa tan solo busca soluciones locales (es decir, se detiene cuando una solución es peor que la anterior), se deberá seleccionar otro que encuentre el mínimo global, que es lo que realmente se busca. Para ello, de entre la gran lista de algoritmos disponibles se elige el algoritmo heurístico Antigone y se debe especificar en el programa como Option MINLP = antigone; que indica que los problemas no lineales de variables enteras-mixtas se resuelvan mediante este algoritmo. La elección de este sistema de resolución se debe a que está destinado a encontrar soluciones globales en problemas con variables binarias y restricciones que impliquen sumatorios. Aunque existen otros algoritmos capaces de resolver este tipo de problemas, se ha elegido el Antigone porque tras las pruebas realizadas ha resultado eficaz y especialmente rápido en encontrar la solución.

Capítulo 5
Modelado del problema determinista

Una vez se han visto cuáles son los diferentes criterios que se han utilizado para resolver el problema, llega el momento de plantear un problema propio que se pueda simular y analizar sus resultados. Siguiendo el orden lógico del desarrollo de las soluciones al problema de asignación de puertas, se va a comenzar por plantear un caso determinista sencillo para después resolver las debilidades encontradas y generar una solución más completa que contemple la mayor cantidad de factores posible. A continuación, se presentará una solución probabilística siguiendo el mismo esquema de desarrollo. Una vez se tengan los dos modelos, se compararán los resultados obtenidos y se extraerán conclusiones acerca de su idoneidad para los distintos escenarios.

Modelo básico: asignación de pasajero a puerta destino

Para familiarizarse con el problema, la nomenclatura y el software utilizado resulta pertinente simular una de las soluciones planteadas en un estudio ya realizado [6] que además contiene un caso numérico real de aplicación. Se debe recordar que esta asignación era muy básica y que una de sus múltiples debilidades era el hecho de asignar pasajeros a puertas y no a vuelos como sería deseable. No obstante, este planteamiento resulta excelente como punto de partida para la generación de un modelo propio que cumpla con los objetivos definidos.

Así pues, siguiendo las fases del modelado se define el problema como:

- Índices:
  - i: vuelo de origen
  - j: puerta de origen
  - k: puerta de destino

- Grupos:
  - F: grupo de vuelos origen (de llegada)
  - G: grupo de puertas disponibles para los vuelos de llegada
  - K: grupo de puertas de destino

- Datos de entrada. Parámetros:
  - $dP_{j,k}$: distancia física entre la puerta j y la puerta k
  - $P_{i,k}$: número de pasajeros del vuelo i que tienen como destino la puerta k

- Datos de salida. Variables del problema:
  - $x_{i,j}$: representa a una variable binaria de valor:

$$x_{i,j} = \begin{cases} 1 & \rightarrow \ si\ el\ vuelo\ i\ está\ asignado\ a\ la\ puerta\ j \\ 0 & \rightarrow \ en\ cualquier\ otro\ caso \end{cases}$$

Estas variables son lo que se espera que devuelva la ejecución del programa. La combinación de ellas debe ser tal que la distancia recorrida total sea mínima.

- Restricciones impuestas:
  - Un vuelo de llegada estará asignado a una y solo a una puerta. Esto se expresa como:

$$\sum_{j \in G} x_{i,j} = 1 \quad \forall i$$

  - Cada puerta vacía podrá albergar como máximo a un vuelo de llegada. Esto se expresa como:

$$\sum_{i \in F} x_{i,j} \leq 1 \quad \forall j$$

- Evaluación del coste de cada solución. Función objetivo: minimizar distancia total.

$$Distancia\ total = \sum_{i \in F} \sum_{j \in G} \sum_{k \in K} P_{i,k} * dP_{j,k} * x_{i,j}$$

Con toda esta información, el modelo queda completamente definido y listo para ponerlo en práctica. Para asegurar que el software utilizado realiza una asignación correcta, se ha decidido utilizar los mismos parámetros de entrada que se utilizan en la publicación [6] . Así, el problema plantea un caso en el que se tienen 7 vuelos de origen, 7 puertas disponibles (puertas de origen) y 19 puertas en total. Todos los datos de pasajeros y distancias se pueden recoger mediante las siguientes matrices de flujo y distancia:

Matriz flujo de pasajeros (n° de pasajeros)																			
Vuelo	Puerta de destino																		
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
F1	5	5	10	8	15	8	2	10	8	20	5	4	0	9	3	4	1	2	5
F2	5	2	1	4	19	9	4	2	3	2	27	3	8	4	0	2	1	7	5
F3	10	0	4	9	13	4	4	4	3	5	5	8	4	9	11	7	9	4	10
F4	4	8	5	4	10	4	1	0	0	2	4	19	1	2	4	5	5	8	4
F5	4	11	9	9	6	3	1	4	4	2	1	0	3	5	1	2	2	3	4
F6	1	2	42	5	2	7	6	2	4	7	2	3	6	4	10	2	1	0	1
F7	3	3	2	5	9	13	11	2	2	3	7	22	4	0	1	1	2	2	3

Tabla 5: Matriz flujo de pasajeros para el modelo básico

Matriz distancias entre puertas (unidades de longitud)																			
Puerta origen	Puerta de destino																		
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	10	40	0	30	10	40	20	50	30	60	40	70	50	80	60	90	70	90	80
4	40	10	30	0	40	10	50	20	60	30	70	40	80	50	90	60	90	70	80
10	70	40	60	30	50	20	40	10	30	0	40	10	50	40	60	30	70	40	50
11	50	80	40	70	30	60	20	50	10	40	0	30	10	40	20	50	30	50	40
14	90	60	80	50	70	40	60	30	50	20	40	10	30	0	40	10	50	20	30
15	70	100	60	90	50	80	40	70	30	60	20	50	10	40	0	30	10	30	20
17	80	100	70	90	60	80	50	70	40	60	30	50	20	40	10	30	0	20	10

Tabla 6: Matriz distancias entre puertas para el modelo básico

Como se observa, la cantidad de variables y ecuaciones es muy grande, por lo que el problema solo se puede resolver mediante técnicas computacionales. Para ello se utiliza la herramienta GAMS ya mencionada anteriormente. El código utilizado es equivalente al modelado matemático realizado y se puede consultar en el anexo. El problema se ha definido como un problema lineal entero-mixto ya que las variables binarias son de naturaleza discreta (solo pueden tomar valores 0 o 1 y no los valores intermedios). Por este motivo se utiliza el comando `solve transport using mip minimizing z;` donde transport es todo el modelo y z es la variable distancia.

Una vez ejecutado el código, el programa devuelve el valor de la distancia que se caminará en el caso óptimo (26000 u. Long.) y el valor de las variables binarias $x_{i,j}$ generadas. Este último se ofrece mediante una tabla:

Solución del problema. Variable $x_{i,j}$							
Vuelo	Puerta asignada						
	3	4	10	11	14	15	17
F1			1				
F2				1			
F3						1	
F4							1
F5		1					
F6	1						
F7					1		

Tabla 7: Solución de organización de vuelos en las puertas para mínima distancia para el modelo básico

Los resultados de la Tabla 7 son equivalentes a decir que el vuelo F1 debe estacionar en la puerta 10, el vuelo F2 en la puerta 11, y así sucesivamente. Se puede observar también que todas las restricciones se cumplen, ya que cada vuelo solo está designado a una puerta, al igual que cada puerta solo tiene a un vuelo designado. Dicha solución coincide con la que se aporta en el artículo [6] por lo que se considera que la herramienta GAMS ha funcionado correctamente para la resolución del problema de la asignación de puertas y se considera validada para la solución de los modelos propios.

Mejora 1: Asignación de pasajero a vuelo de destino

Dado que la asignación anterior tenía muchas limitaciones como ya se ha comentado en el estado del arte, parece interesante implantar una serie de mejoras para que sea un sistema funcional. Así pues, la primera implementación pasa por cambiar la forma de entrada de los datos para que sea como lo es en la realidad: los pasajeros de cada vuelo tendrán como destino otro de los vuelos. Este hecho hace que se deba añadir una segunda variable de decisión que permita contemplar de forma independiente la puerta de origen y la de destino a la hora de calcular la distancia. Con ello, si se sigue un esquema igual al del modelo básico descrito, se debería introducir un cuarto índice que haga referencia a la variable puerta de destino.

Para tener un sistema cuya cantidad de variables sea asumible de forma que el problema tenga solución, se decide elaborar una forma propia de computar las distancias entre puertas que utilice un solo índice. Esta forma consiste en considerar las distancias entre puertas como diferencia entre distancias entre cada puerta y un punto de origen. De esta forma, tan solo existirá un índice referido a las puertas y los datos de distancia se introducirán como vector y no como matriz. En el siguiente grafico se puede ver un ejemplo de cómo se calcula esta separación.

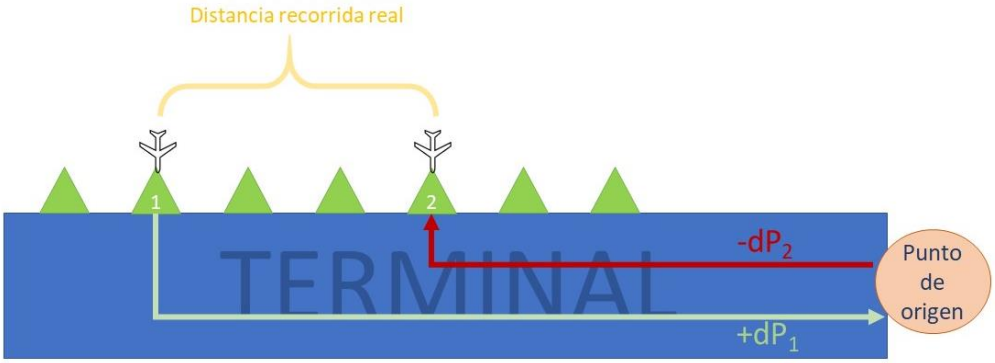


Ilustración 3: Ejemplo de la computación matemática de la distancia recorrida por los pasajeros de la puerta 1 a la puerta 2

En el ejemplo de la Ilustración 3 se observa cómo para introducir en el modelo la distancia recorrida real entre la puerta 1 y la puerta 2, se considera que el pasajero viaja hasta el punto de origen (línea verde) y después vuelve hasta la puerta de destino que le corresponde (línea roja). Dado que lo que interesa al cálculo es la distancia real, se introduce el segundo término como negativo. Después, para asegurar el correcto funcionamiento del programa se debe tomar el valor absoluto de la diferencia de distancias puesto que la puerta de origen podría estar más cerca del punto de origen que la puerta de destino. Así la distancia recorrida por los pasajeros queda como $Distancia\ recorrida_{1\to2} = número\ pasajeros_{1\to2} * |dP_1 - dP_2|$.

Una vez comprendida esta asignación es fácil comprender el modelo propuesto. En él, se define la siguiente notación:

- Índices:
  - i: vuelo de origen
  - j: vuelo de destino

- *puert*: puerta de embarque

- Grupos:
  - F: grupo de vuelos origen (de llegada)
  - G: grupo de vuelos destino (de salida)
  - K: grupo de puertas del aeropuerto

- Parámetros:
  - dP_{puert}: distancia física entre la puerta *puert* y el origen
  - $P_{i,j}$: número de pasajeros del vuelo i que tienen como destino el vuelo j

- Función objetivo: minimizar distancia total:

$$Distancia\ total = \sum_{i \in F} \sum_{j \in G} P_{i,j} * \left| \sum_{puert \in K} dP_{puert} * x_{i,puert} - \sum_{puert \in K} dP_{puert} * y_{j,puert} \right|$$

Donde $x_{i,puert}$ representa a una variable binaria de valor:

$$x_{i,puert} = \begin{cases} 1 & \rightarrow \ si\ el\ vuelo\ i\ está\ asignado\ a\ la\ puerta\ puert \\ 0 & \rightarrow \ en\ cualquier\ otro\ caso \end{cases}$$

Y de forma análoga, $y_{j,puert}$ representa a una variable binaria de valor:

$$y_{j,puert} = \begin{cases} 1 & \rightarrow \ si\ el\ vuelo\ j\ está\ asignado\ a\ la\ puerta\ puert \\ 0 & \rightarrow \ en\ cualquier\ otro\ caso \end{cases}$$

- Restricciones impuestas:

  - Un vuelo de llegada estará asignado a una y solo a una puerta. Esto se expresa como:

$$\sum_{puert \in K} x_{i,puert} = 1 \quad \forall i$$

  - Un vuelo de salida estará asignado a una y solo a una puerta. Esto se expresa como:

$$\sum_{puert \in K} y_{j,puert} = 1 \quad \forall j$$

  - Una puerta albergará a uno o ningún vuelo de origen. Dado que se trabaja con variables binarias, esto se expresa como:

$$\sum_{i \in F} x_{i,puert} \leq 1 \quad \forall puert$$

- Una puerta albergará a uno o ningún vuelo de destino. Esto se expresa como:

$$\sum_{j \in G} y_{j,puert} \leq 1 \quad \forall puert$$

- Si se desea mantener la nomenclatura entre vuelos de origen y de destino (es decir, que el vuelo i de origen y el vuelo j de destino tengan la misma puerta) se debe cumplir que:

$$x_{i,puert} = y_{j=i,puert} \quad \forall i, puert$$

Para comprender cómo funciona el modelo propuesto resulta muy oportuno introducir un ejemplo con datos inventados. En este caso, los datos de entrada son la matriz de flujo de pasajeros (esta vez entre vuelos) y el vector de distancias de las puertas hasta el origen. Se observa que en este caso se ha hecho coincidir que el origen de distancias de las puertas es en P3, por lo que la distancia entre este punto y la puerta es 0. Los datos que se han utilizado en la simulación son:

Matriz flujo de pasajeros (n° de pasajeros)			
Vuelo de origen	Vuelo de destino		
	V1	V2	V3
V1	0	50	30
V2	0	0	0
V3	0	0	0

Tabla 8: Matriz flujo de pasajeros para la mejora 1

Vector distancias de la puerta al punto de origen (unidades de longitud)		
P1	P2	P3
1000	800	0

Tabla 9: Vector distancias de la puerta al punto de origen para la mejora 1

El código utilizado traduce el modelo al lenguaje de la herramienta y se puede consultar en el anexo. Una vez ejecutado el programa, se devuelve una matriz con el valor de las variables binarias generadas. Lo primero que se puede observar es que tanto la variable $x_{i,puert}$ como la $y_{j,puert}$ tienen el mismo valor, como se había especificado en las restricciones. Dicho valor se ha asignado de forma que la distancia total es de 34000 u. Long. La solución por tanto es:

Solución del problema. Variable $x_{i,puert}$=$y_{j,puert}$			
Vuelo	Puerta asignada		
	P1	P2	P3
V1		1	
V2	1		
V3			1

Tabla 10: Solución de organización de vuelos en las puertas para mínima distancia para la mejora 1

Dados los datos introducidos, la solución era esperable y se podía observar a simple vista que esa sería la configuración óptima. Un gráfico ayuda a observar la solución:

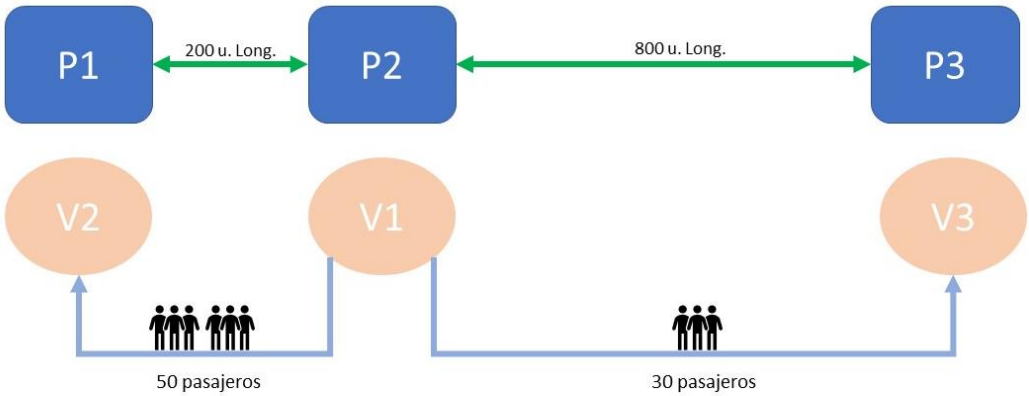


Ilustración 4: Gráfico representativo de la solución del ejemplo de la mejora 1

Como se puede observar, con los datos de la matriz de flujo resulta evidente que el vuelo V1 debe estar en el centro y que V2 debe estar en la puerta más cercana ya que es al que más pasajeros se dirigen.

A pesar de que la solución para este caso es correcta si se utiliza el *solver* por defecto del programa, se observa que al realizar pequeñas variaciones la solución aportada no siempre es la óptima pese a que el cálculo de la distancia recorrida sí es correcto. Estas variaciones consisten en realizar cambios menores que no deberían alterar el resultado, como añadir un solo pasajero del vuelo que de V2 se dirige a V3. Esto sucede porque el programa se detiene cuando en sus iteraciones encuentra una solución con distancia total mayor que la anterior. En lenguaje del programa: la solución está empeorando. No obstante, esto se debe a que se trata de un mínimo local de la función y no del mínimo global, que es lo que se busca. Para solucionarlo, se ha tenido que cambiar el *solver* utilizado e introducir uno que tenga en cuenta la existencia de otros mínimos locales.

Tras una investigación sobre los muchos algoritmos de resolución, se ha elegido el ANTIGONE ya que es el que mejor encaja con las características del problema (búsqueda del mínimo global de un minlp con variables mixtas). Al ejecutar de nuevo el programa con este *solver* se encuentra la solución óptima, tanto para el problema tal y como se ha definido como para las pequeñas variaciones introducidas a modo de prueba.

Una vez se ha observado que el modelo es correcto y ofrece realmente la solución óptima, se va a mostrar un ejemplo realizado con datos también inventados, pero de mayor complejidad. Para ello se parte de los siguientes datos:

Matriz flujo de pasajeros (n° de pasajeros)															
V. de origen	Vuelo de destino														
	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15
V1	15	19	0	0	0	0	0	0	0	0	0	0	27	0	0
V2	28	0	0	7	0	0	32	0	13	0	0	0	0	0	0
V3	0	0	0	0	0	12	0	12	0	43	0	0	0	0	0
V4	0	0	10	12	0	0	0	0	0	0	0	0	0	0	0
V5	0	0	0	23	0	0	8	0	0	23	0	0	12	0	0
V6	0	0	0	0	23	0	8	0	0	0	0	23	0	15	0
V7	0	0	0	0	0	45	0	0	0	0	0	0	27	23	17
V8	0	0	0	0	17	0	0	21	0	0	0	0	0	0	0
V9	0	0	0	24	28	0	13	0	0	0	32	0	0	0	24
V10	0	0	40	0	0	0	0	0	0	0	0	0	0	23	0
V11	12	25	0	0	0	0	0	0	12	0	0	34	0	0	0
V12	0	0	12	0	0	0	0	0	0	0	0	0	0	0	0
V13	0	0	0	21	0	0	0	0	0	0	0	0	12	0	0
V14	0	0	0	0	45	0	0	0	12	0	25	0	0	0	0
V15	12	0	0	12	0	0	9	10	0	22	0	0	0	0	0

Tabla 11: Matriz flujo de pasajeros para la mejora 1 en el caso complejo

Vector distancias de la puerta al punto de origen (unidades de longitud)					
P1	3020	P8	1680	P15	350
P2	2870	P9	1450	P16	270
P3	2690	P10	1210	P17	120
P4	2410	P11	1020	P18	180
P5	2200	P12	890	P19	50
P6	2060	P13	640	P20	2
P7	1830	P14	420		

Tabla 12: Vector distancias de la puerta al punto de origen para la mejora 1 en el caso complejo

Se observa que se trata de un caso que va a tener muchas opciones ya que no solo existen 15 vuelos, sino que existen 20 puertas, de forma que 5 de ellas se quedarán vacías. Cabe destacar que la opción más lógica será dejar todos los huecos juntos en uno de los extremos. Introduciendo estos datos en el programa se obtiene la siguiente asignación:

Solución del problema. Variable $x_{i,puert}$=$y_{j,puert}$															
Vuelo	Puerta asignada														
	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20
V1															1
V2														1	
V3		1													
V4					1										
V5						1									
V6									1						
V7										1					
V8	1														
V9							1								
V10			1												
V11													1		
V12												1			
V13											1				
V14								1							
V15				1											

Tabla 13: Solución de organización de vuelos en las puertas para mínima distancia para la mejora 1 en el caso complejo

Se obtiene también que la distancia total recorrida por los pasajeros para este caso es de 390834 u. Long. Aunque no se puede comprobar de forma lógica si realmente es la solución óptima, sí se ha comprobado que realizando cambios puntuales entre dos puertas las distancias siempre aumentaban.

Mejora 2: Añadir horarios de llegada y salida

Una vez se ha comprobado que el modelo que asigna pasajeros con vuelos funciona correctamente, se puede añadir la siguiente característica: posibilidad de que dos vuelos utilicen una misma puerta siempre y cuando sus horarios no se solapen. Para añadir esta característica, la función objetivo, la definición de variables y la mayoría de las restricciones impuestas se mantienen. El principal cambio en el modelo es la inclusión de dos nuevos vectores (parámetros) que contienen la información relativa al minuto de llegada y de salida del vuelo. Estos se definen como:

- $a_i = a_j$: hora de llegada del vuelo i (o j). Nótese que, al aplicar la restricción que obliga a que vuelos con el mismo nombre tengan la misma puerta, ambos vectores son iguales.
- $d_i = d_j$: hora de salida del vuelo i (o j).

De igual forma se deben sustituir las restricciones referentes al "máximo un avión por puerta" por una que contemple los horarios de estos. Así se introduce la nueva restricción:

$$x(i,puert) * y(j,puert) * |a_i - a_j| * (d_j - a_i) * (d_i - a_j) \leq 0 \qquad \forall i,j,puert$$

Esto implica que para cada combinación de i,j,puert se pueden dar dos casos en los que se cumple la restricción:

- Ecuación $0 \leq 0$: si alguno de los tres primeros términos es igual a 0. Esto implica que si una combinación de vuelos i-j no está asignada a la puerta puert, la ecuación es cero. El término contenido en el valor absoluto asegura que la ecuación también será 0 en los casos en que $i = j$ ya que, en este caso, sí se permite que ambos compartan puerta (son el mismo vuelo).
- $Término\ negativo \leq 0$: si los vuelos considerados están ubicados en la puerta puert y tienen horarios que no solapan. El hecho de incluir el producto de dos términos se debe a que no se sabe cuál es la posición relativa de los vuelos. Para comprender correctamente esto se incluye un gráfico ilustrativo de las dos situaciones posibles:

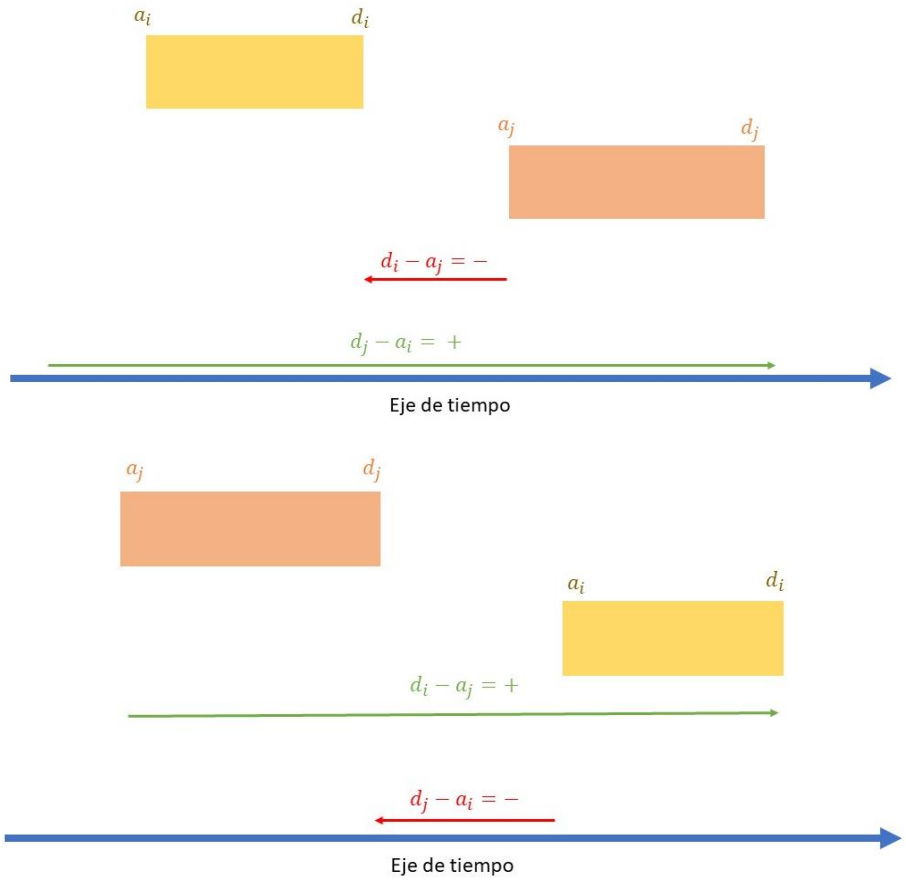


Ilustración 5: Diagrama de las posibilidades de colocación de dos vuelos que podrían compartir puerta

Como se observa, se necesita que cualquiera de los dos términos sea negativo para que la asignación sea válida. Si ambas fueran positivas significaría que ambos vuelos se solapan y por tanto no sería una asignación válida.

Como punto negativo de este modelo, se encuentra que, si existen **dos vuelos con idéntica hora de llegada**, el sistema considerará valido asignar ambos a una misma puerta (el producto de $|a_i - a_j|$ será 0 sin tratarse de un mismo vuelo) cuando se sabe que esto es correcto porque son vuelos distintos que se solapan. Una posible forma de evitar esta anomalía sería introducir un término más con la diferencia de horas de salida. No obstante, se decide imponer como condición a los datos de entrada que si se da este caso se deberán introducir las horas de

llegada como números decimales distintos (por ejemplo, dos vuelos llegando en el minuto 50 se catalogarán como $a_1 = 50.5$ y $a_2 = 50$).

A continuación, se plantea un ejemplo sencillo que comprueba que el modelo funciona correctamente. Los datos de entrada son los siguientes:

Matriz flujo de pasajeros (n° de pasajeros)					
Vuelo de origen	Vuelo de destino				
	V1	V2	V3	V4	V5
V1	0	50	30	50	30
V2	0	0	0	0	0
V3	0	0	0	0	0
V4	0	0	0	0	0
V5	0	0	0	0	0

Tabla 14: Matriz flujo de pasajeros para la mejora 2

Vector distancias de la puerta al punto de origen (unidades de longitud)		
P1	P2	P3
1000	800	0

Tabla 15: Vector distancias de la puerta al punto de origen para la mejora 2

Vectores hora de llegada y hora de salida (minutos desde el origen)		
Vuelo	Hora de llegada (a)	Hora de salida (d)
V1	0	40
V2	10	50
V3	15	55
V4	50	120
V5	55	120

Tabla 16: Vectores hora de llegada y hora de salida para la mejora 2

Como se observa, el escenario es similar al planteado para la mejora 1, solo que ahora hay cinco vuelos para las mismas tres puertas. No obstante, cuando llegan los últimos, los primeros ya se habrán marchado.

Una vez resuelto el problema, se encuentra que la solución óptima implica una distancia total de 40000 unidades de longitud. La asignación de puertas asociada es:

Solución del problema. Variable $x_{i,puert}$=$y_{j,puert}$			
Vuelo	Puerta asignada		
	P1	P2	P3
V1		1	
V2	1		
V3			1
V4		1	
V5	1		

Tabla 17: Solución de organización de vuelos en las puertas para mínima distancia para la mejora 2

Como se observa, la asignación de los tres primeros vuelos es idéntica a la de la mejora 1 ya que los datos de pasajeros y distancias son los mismos. Después, los vuelos V1 y V5 se asignan a las puertas más cercanas a P2, que es de donde parten los pasajeros, teniendo en cuenta que el número mayor de pasajeros es el que va a V4 (y por ello queda en la misma puerta).

Capítulo 6
Modelado del problema probabilista

Una vez estudiados los casos deterministas llega el turno de crear un modelo propio que tenga en cuenta la posibilidad de retrasos y variaciones en los horarios de los vuelos. Como en el caso determinista, existen muchas formas de interpretar los datos y distintas variables que optimizar. Para el estudio se decide utilizar una asignación en dos fases que minimice la posibilidad de concurrencia de más de una aeronave en la misma puerta en primera instancia y, después, que asigne la puerta física concreta de cada asignación. La primera fase necesitará de los datos de la hora de llegada y salida de cada vuelo, así como de las desviaciones típicas asociadas a cada una. Con esto y teniendo en cuenta el número de puertas disponibles se asigna los vuelos a las puertas de forma que se minimice lo que se ha llamado riesgo de conflicto global. Esto no es más que la suma de las probabilidades de la presencia de las aeronaves asignadas a la puerta para cada una de las franjas horarias contempladas. Como se puede observar, esta asignación no contempla ninguna característica física de la puerta y tendrá muchas soluciones óptimas, ya que lo único que se tiene en cuenta es la combinación de vuelos que alberga. Esto significa que la función a optimizar tendrá el mismo resultado si se asigna la combinación de vuelos V1 y V2 a la puerta P1 o P2. Por este motivo, en esta fase no se asignarán puertas sino horarios (H). Así, una segunda fase del modelo buscará optimizar dicha asignación para que, teniendo en cuenta todos los vuelos asignados a una misma puerta, se asigne una puerta física que minimice la distancia total que caminan los pasajeros de forma similar a la observada en los casos deterministas.

Dado que se trata de un estudio sobre datos probabilistas es necesario determinar qué función de probabilidad seguirán las horas de llegada y salida de los vuelos. Como no se dispone de datos reales, no se puede estudiar la correlación de la realidad con las distintas funciones conocidas y por ello se decide utilizar una distribución normal definida por su media y varianza. Para introducir estos datos en el estudio se van a utilizar valores al azar y ya definidos para el vuelo en concreto. No obstante, en un escenario real en el que se disponga de una cantidad abundante de datos se debería realizar una correlación cruzada entre distintos parámetros de los vuelos que pueden afectar a la puntualidad de los vuelos. Algunos de los parámetros que se podría utilizar son: aerolínea, origen, destino, temporada (verano, invierno), día de la semana, etc.

Fase 1: Mínimo riesgo de conflicto global

Esta parte es la que considera los tiempos de llegada y salida de los vuelos para asignarlos a uno de los horarios disponibles, H (igual al número de puertas). Lo primero que se debe comprender es cómo se introducen los datos en el sistema y qué es lo que representan. Para cada uno de los vuelos se debe incluir la siguiente información: hora media de llegada, desviación estándar de la llegada, hora media de salida y desviación estándar de la salida. Con todo esto se generan dos funciones, una función acumulativa de distribución de probabilidad y una acumulativa inversa, ambas siguiendo la distribución normal. La primera representa la

probabilidad que existe de que el vuelo haya llegado a la puerta en un determinado minuto y la segunda la probabilidad de que el vuelo haya salido. Nótese que la función acumulativa inversa equivale a uno menos la función acumulativa.

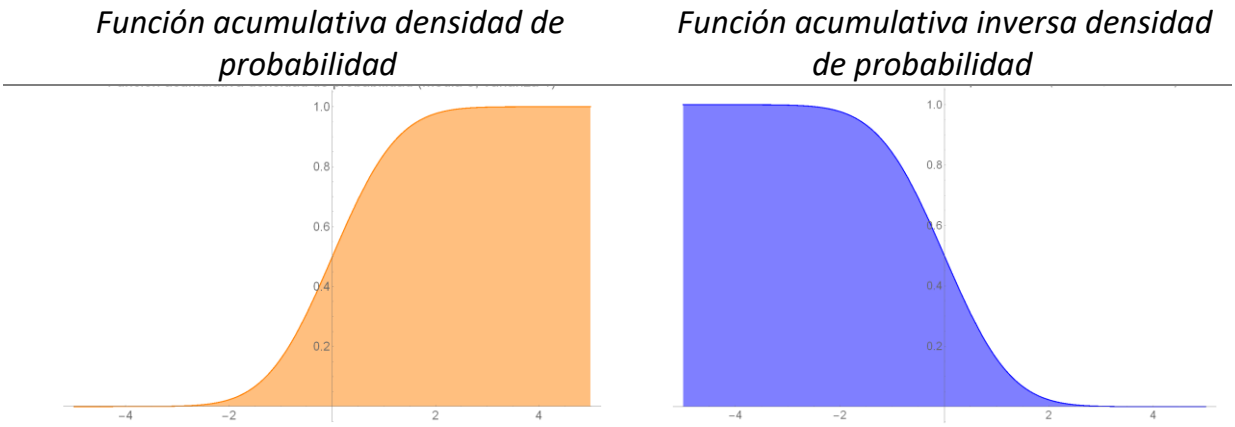


Ilustración 6: Funciones acumulativas densidad de probabilidad de media 0 y desviación estándar 1

De esta forma, si considera la intersección de las dos funciones, se tiene la función de distribución de probabilidad de que el avión esté en la puerta en un determinado momento.

Una vez comprendido cómo se representa el hecho de que la aeronave esté en la puerta, es el momento de centrarse en la optimización que se va a realizar. De forma general la probabilidad de que dos hechos sucedan equivale al producto de probabilidades. Así, para dos vuelos que estén asignados en una misma puerta, la probabilidad de que exista concurrencia equivale al producto de sus respectivas funciones de probabilidad en un determinado instante. Si se desea conocer la probabilidad de conflicto global de una puerta que contiene dos aviones se debería evaluar la integral del producto de sus funciones. No obstante, un modelado que incluyera esta opción resultaría muy complejo y costoso de resolver por lo que se utiliza una aproximación a dicho resultado. Dicha aproximación consiste en evaluar el sumatorio del producto de las funciones de probabilidad de presencia del avión para cada uno de los minutos de la franja temporal estudiada. Evidentemente, el resultado de esta operación no equivale a la probabilidad de conflicto en la puerta evaluada mediante la integral; no obstante, sí permite encontrar la solución cuyo resultado sería el menor, que es el objetivo último del problema.

Así, se propone la siguiente notación:

- Índices:
  - i: vuelos que deben ser atendidos
  - j: vuelos que deben ser atendidos
  - hor: horario de puerta
  - *minute*: minutos de la simulación

Las funciones necesitarán el producto de parámetros de dos vuelos, por lo que resulta útil definir dos índices referidos a los vuelos atendidos.

- Grupos:
  - F: grupo de vuelos

- L: grupo de horario de puerta (mismo número que de puertas disponibles)
- M: grupo de minutos de la simulación

- Parámetros:
  - $mediaa_i$: valor medio de la función de probabilidad de minuto de llegada
  - $desva_i$: desviación estándar de la función de probabilidad de minuto de llegada
  - $mediad_i$: valor medio de la función de probabilidad de minuto de salida
  - $desvd_i$: desviación estándar de la función de probabilidad de minuto de salida

Todos estos datos permiten obtener el que realmente es el parámetro de la función de optimización:

- $prob_{minute,i}$: valor de la función de probabilidad de presencia de la aeronave *i* en su puerta en el minuto *minute*.
- $prob_{minute,j}$: dado que i y j se refieren a unos mismos vuelos, los parámetros $prob$ serán idénticos. $prob_{minute,j} = prob_{minute,i=j}$

- Función objetivo: minimizar el riesgo de conflicto total:

$$Riesgo\ total = \sum_{minute \in M} \sum_{hor \in L} \sum_{i \in F} \sum_{j>i} (x_{i,hor} * prob_{minute,i}) * (y_{j,hor} * prob_{minute,j})$$

Donde $x_{i,hor}$ representa a una variable binaria de valor:

$$x_{i,hor} = \begin{cases} 1 & \rightarrow \ si\ el\ vuelo\ i\ está\ asignado\ al\ horario\ hor \\ 0 & \rightarrow \ en\ cualquier\ otro\ caso \end{cases}$$

Y $y_{j,hor}$ tiene un valor análogo.

Esta expresión equivale al sumatorio del producto de las funciones de probabilidad de presencia del avión para cada uno de los minutos de la franja temporal estudiada. Esto es evaluar la probabilidad de que dos aeronaves que estén asignadas al mismo horario coincidan (por eso se incluyen las variables binarias) para cada minuto de cada horario. Así, si solo hay un avión en el horario, el riesgo de conflicto es cero.

Dado que lo que se quiere es el producto de todas las combinaciones de vuelos y se tienen dos variables distintas, se debe especificar que el segundo sumatorio solo debe atender las $j > i$. Comprender este elemento resulta muy sencillo si se observa un ejemplo sencillo de solo cuatro vuelos. Si en vez de atender en el segundo sumatorio a las $j > i$ se atendiera a todas las j contenidas en F se obtendrían todos los valores de la Tabla 18. Por tanto, dado que los valores i y j son equivalentes, se obtiene el producto de cada combinación dos veces (propiedad conmutativa). Así, cuando se tienen en cuenta solo los valores del segundo índice (en este caso el j) mayores que el primero se obtienen las combinaciones deseadas. Resta decir que cuando los índices son $i = j$, el producto carece de sentido ya que equivale a multiplicar la probabilidad de presencia de un avión consigo mismo, lo cual no debería añadir ningún término a la ecuación de riesgo global.

Valor de i	i1	i2	i3	i4
Combinaciones posibles	i1*j1	i2*j1	i3*j1	i4*j1
	i1*j2	i2*j2	i3*j2	i4*j2
	i1*j3	i2*j3	i3*j3	i4*j3
	i1*j4	i2*j4	i3*j4	i4*j4

Tabla 18: Distintas combinaciones de valores i y j para ejemplificar los índices en la función objetivo del modelo probabilístico de fase 1. Los valores en verde son los que se desea obtener

- Restricciones impuestas:
  - Un vuelo estará asignado un solo a un horario. Esto se expresa como:

$$\sum_{hor \in L} x_{i,hor} = 1 \quad \forall i$$

  - Los vuelos con mismo índice deben estar asignados al mismo horario. Esto se expresa como:

$$x_{i,hor} = y_{j=i,hor} \quad \forall hor, i$$

A continuación, se plantea un ejemplo sencillo que comprueba que el modelo funciona correctamente. Se dispone de un aeropuerto con **tres puertas** por lo que existen tres horarios distintos que establecer. Los otros datos de entrada son los siguientes:

Vectores media de hora de llegada y hora de salida (minutos desde el origen) y desviaciones estándar				
Vuelo	Hora de llegada (a)		Hora de salida (d)	
	Media	Desviación estándar	Media	Desviación estándar
V1	90	5	120	5
V2	110	10	230	10
V3	125	20	245	10
V4	160	20	250	30
V5	175	15	300	20

Tabla 19: Vectores media de hora de llegada y hora de salida (minutos desde el origen) y desviaciones estándar para la fase 1

Una vez resuelto el problema, se encuentra que la solución óptima implica un riesgo de conflicto global de 25.883. La asignación de puertas asociada es:

Solución del problema. Variable $x_{i,hor}$=$y_{j,hor}$			
Vuelo	Horario asignado		
	H1	H2	H3
V1		1	
V2	1		
V3			1
V4		1	
V5	1		

Tabla 20: Solución de organización de vuelos en horarios para mínimo conflicto global en la fase 1

Para comprender cómo funciona el modelo de probabilidades propuesto resulta útil representar gráficamente las funciones de probabilidad de presencia de aquellos vuelos que comparten horario. Dados los datos y la solución dada, las probabilidades en cada horario para los minutos de 0 a 350 se tiene:

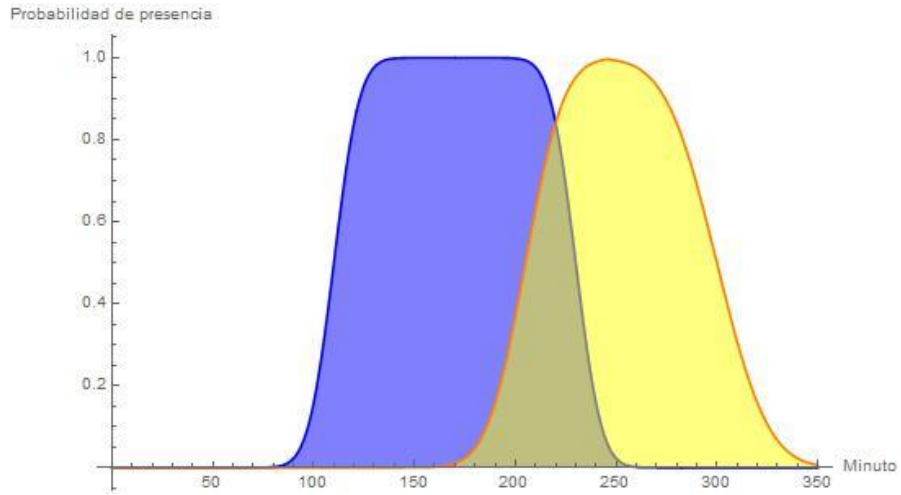


Ilustración 7: Probabilidades de presencia en el horario H1 (V2 en azul y V5 en amarillo)

Ilustración 8: Probabilidades de presencia en el horario H2 (V1 en verde y V4 en naranja)

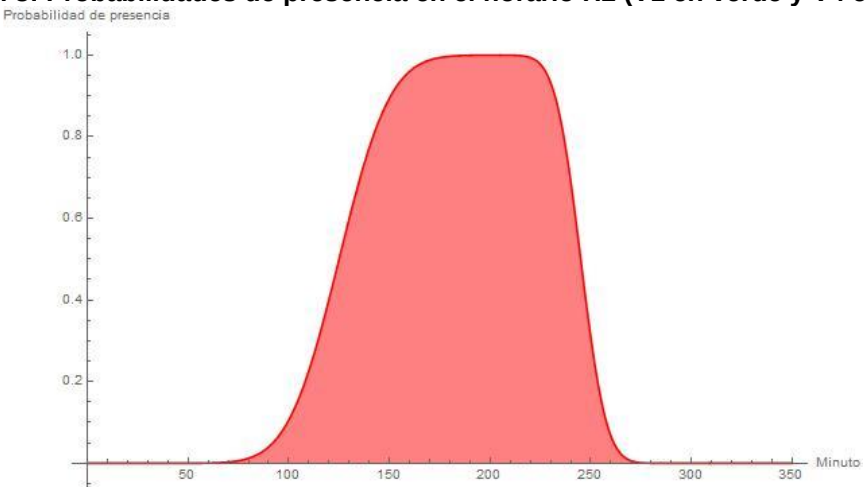


Ilustración 9: Probabilidades de presencia en el horario H3 (V3)

Se tiene por tanto una asignación que, dada la cantidad de tres puertas, minimiza el total de las áreas sombreadas por los vuelos, es decir, las intersecciones. Se observa que en este

ejemplo la probabilidad de conflicto va a ser muy alta, ya que el área conjunta correspondiente al H1 es muy grande. Para comprobar que efectivamente la probabilidad de conflicto de la asignación realizada es la menor, se puede ejecutar mediante Mathematica el área real correspondiente a la intersección de las curvas mediante la integral. Como es fácil observar, lo óptimo será que el horario que contenga a un solo lo vuelo lo haga con V3 ya que es el que está en el centro. Por tanto, se va a comparar el resultado de la asignación dada por el programa con la asignación de V1 con V5 y V2 con V4, ya que se observa a simple vista que cualquier otra combinación tendrá mayor riesgo. La Tabla 21 muestra como efectivamente el programa ha dado la solución de menor riesgo global.

Intersección V1-V4	**0.596**	Total: 93.513
Intersección V2-V5	92.917	
Intersección V1-V5	30.003	Total: 100.589
Intersección V2-V4	70.586	

Tabla 21: Valores de la integral de la intersección de distintas combinaciones de vuelos

Fase 2: Mínima distancia andada por los pasajeros

Una vez se ha asignado qué vuelos formarán cada horario, se ha visto que desde el punto de vista de la probabilidad de conflicto, es indiferente asignar el horario a una puerta u otra. No obstante, una vez se ha generado un sistema robusto, se puede utilizar el "grado de libertad" que queda para minimizar en la medida de lo posible la distancia que deben andar los pasajeros. Para ello, se va a utilizar un modelo muy parecido al que se ha visto en la Mejora 1: Asignación de pasajero a vuelo de destino. La principal diferencia reside en que ahora en vez de utilizar los índices i y j para referirse a vuelos de origen y destino, se utilizan los índices i_hor y j_hor para referirse a los horarios de origen y destino. De esta forma, previamente a la introducción de los datos en el programa, habrá que sumar y agrupar los pasajeros que compartan horario de origen y se dirijan hacia un mismo horario de destino. Esta es la información que se introduce en la matriz P_{i_hor,j_hor}, mientras que el vector dP_{puert} se mantiene igual.

El funcionamiento de esta asignación se observa de forma muy clara si se realiza la asignación de puertas a los horarios de la primera fase observados en la Tabla 20. Observando la agrupación realizada y suponiendo que los pasajeros tienen mismos orígenes y destinos que los utilizados en el ejemplo de la mejora 2 y que se pueden ver en la Tabla 14, se tienen los siguientes datos:

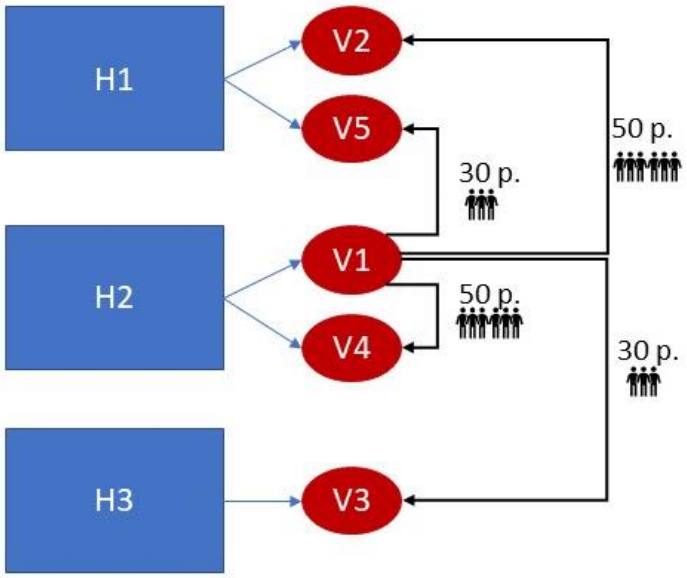


Ilustración 10: Datos iniciales modelo probabilista fase 2

Esto se deberá introducir en el programa como:

Matriz flujo de pasajeros (n° de pasajeros)			
Horario de origen	Horario de destino		
	H1	H2	H3
H1	0	0	0
H2	80	50	30
H3	0	0	0

Tabla 22: Matriz flujo de pasajeros para el modelo probabilista en la fase 2

Dado que este caso es muy sencillo, se puede observar a simple vista cómo quedará la matriz de flujo de pasajeros entre los horarios. No obstante, en casos mucho más complejos el cálculo de esta matriz puede ser muy tedioso y largo, por ello se ha diseñado un pequeño programa en Matlab que recorre la matriz de flujos de los vuelos y los asigna a su combinación de horario de origen – horario de destino correspondiente según la tabla solución de la fase 1. El código correspondiente se puede observar en el anexo.

De igual forma, se debe introducir una matriz de distancia entre puertas. Dado que este caso solo necesita de tres puertas (debe ser el mismo número que de horarios), se decide utilizar el mismo vector que el utilizado en Mejora 1: Asignación de pasajero a vuelo de destino, y que se observa en la Tabla 9.

Una vez introducida la información en el modelo, la solución dada es igual a la que nos ofrece la lógica observando unos datos tan sencillos como los utilizados. En este caso el programa nos da una distancia total de 40000 unidades de longitud.

Solución del problema. Variable $x_{i\_hor,puert}$=$y_{j\_hor,puert}$			
Horario	Puerta asignada		
	P1	P2	P3
H1	1		
H2		1	
H3			1

Tabla 23: Solución de organización de horarios en las puertas para mínima distancia en la fase 2

Dicha asignación a su vez equivale a decir que los vuelos V2 y V5 están asignados a P1, los vuelos V1 y V4 a P2 y el vuelo V3 a la puerta P3.

Capítulo 7
Comparativa de resultados

Como parte final del estudio, resulta útil comparar los resultados obtenidos en ambos métodos para observar sus potenciales ventajas e inconvenientes. Para ello, se ha realizado un estudio a gran escala (con 50 vuelos) con los mismos datos de entrada en ambos casos. El principal objetivo será por tanto comparar la distancia total que deberán andar los pasajeros y observar cuál es "el precio a pagar" por tener un sistema totalmente robusto frente a uno que no lo es.

Dado que no ha sido posible obtener datos reales con toda la información necesaria, se ha propuesto un caso completamente inventado con 50 vuelos a distintas horas para un aeropuerto de 16 puertas en las que todas ellas pueden ser utilizadas indistintamente por cualquiera de los vuelos. La configuración del aeropuerto sigue la configuración más sencilla posible: una sola terminal lineal estándar con todas las puertas al mismo lado y a distintas distancias entre sí.

Esta configuración equivale a un vector de distancias, que es el que se introducirá en programa, como:

Vector distancias de la puerta al punto de origen (m)			
P1	500	P9	250
P2	450	P10	180
P3	420	P11	150
P4	400	P12	130
P5	380	P13	100
P6	350	P14	70
P7	300	P15	40
P8	270	P16	0

Tabla 24: Vector distancias de la puerta al punto de origen para la comparación

A su vez, los vuelos se establecen mediante los siguientes horarios de llegada y salida con su correspondiente varianza:

Vectores media de hora de llegada y hora de salida (minutos desde el origen) y desviaciones estándar									
	Minuto de llegada (a)		Minuto de salida (d)			Minuto de llegada (a)		Minuto de salida (d)	
	Media	Desv. Est.	Media	Desv. Est.		Media	Desv. Est	Media	Desv. Est.
V1	10	25	125	10	V26	400	4	490	11
V2	80	40	170	3	V27	410	6	470	14
V3	90	12	190	4	V28	415	3	510	8
V4	130	8	255	6	V29	405	7	550	7
V5	200	4	290	8	V30	370	7	495	6
V6	205	9	260	1	V31	420	5	550	3
V7	220	14	310	4	V32	435	6	500	4
V8	225	20	335	3	V33	460	15	585	8
V9	230	10	290	9	V34	439	35	555	5
V10	245	42	382	8	V35	470	4	590	4
V11	255	14	365	5	V36	478	16	610	3
V12	284	23	320	3	V37	466	8	645	7
V13	290	4	410	5	V38	493	9	690	8
V14	268	8	415	14	V39	490	7	611	10
V15	295	9	390	22	V40	500	6	685	1
V16	287	6	320	17	V41	530	13	649	3
V17	302	23	425	16	V42	524	23	642	6
V18	320	10	450	11	V43	532	26	694	8
V19	335	11	400	14	V44	527	18	614	11
V20	330	7	460	5	V45	540	19	680	4
V21	315	8	355	8	V46	550	5	700	5
V22	355	6	435	7	V47	594	7	750	6
V23	346	8	400	6	V48	574	26	710	10
V24	365	15	501	6	V49	590	12	698	8
V25	380	8	560	5	V50	603	6	713	3

Tabla 25: Vectores media de hora de llegada y hora de salida (minutos desde el origen) y desviaciones estándar para la comparación

Por último, se debe especificar la matriz de pasajeros entre los vuelos. Esta, se ha establecido de forma que en cada vuelo llegan algunos pasajeros cuyo destino son vuelos que saldrán más tarde.

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16	V17	V18	V19	V20	V21	V22	V23	V24	V25
V1	0	25	18	3	7	0	22	0	0	3	0	9	14	0	0	0	0	0	0	0	0	0	0	0	0
V2	0	0	0	12	19	4	8	0	22	18	0	4	0	5	0	0	0	0	0	0	0	0	0	0	0
V3	0	0	0	6	7	24	0	54	0	0	6	0	0	1	0	7	0	0	0	0	0	0	0	0	0
V4	0	0	0	0	0	0	8	19	34	4	0	0	8	0	2	0	12	0	0	0	0	0	0	0	0
V5	0	0	0	0	0	0	10	0	20	6	3	1	0	7	0	15	0	10	0	0	0	0	0	0	0
V6	0	0	0	0	0	0	1	12	0	7	2	9	0	4	0	0	15	0	0	0	0	0	0	0	0
V7	0	0	0	0	0	0	0	0	0	10	6	7	22	19	7	0	8	0	4	0	10	0	0	0	0
V8	0	0	0	0	0	0	0	0	0	1	1	0	7	7	0	1	0	23	0	4	22	0	0	0	0
V9	0	0	0	0	0	0	0	0	0	0	7	10	8	8	0	4	6	0	10	8	0	1	0	0	0
V10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	6	20	2	2	7	14	13	0
V11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	0	7	3	0	1	8	2	14
V12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	10	7	4	3	0	0	4	0
V13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	7	30	0	19	0	0	0
V14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	8	0	0	4	0	9
V15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	2	3	7	3	0	4	0
V16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	8	0	20
V17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	3
V18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0
V19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	0
V20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	4
V21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0
V22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0
V23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	2

	V26	V27	V28	V29	V30	V31	V32	V33	V34	V35	V36	V37	V38	V39	V40	V41	V42	V43	V44	V45	V46	V47	V48	V49	V50
V10	11	3	0	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V11	26	7	4	4	7	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V12	20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V13	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V14	0	7	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V15	10	0	20	5	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V16	0	4	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V17	4	2	17	5	3	4	4	0	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V18	8	6	3	5	1	6	2	7	6	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V19	2	20	0	3	7	0	0	0	0	10	0	8	2	0	0	0	0	0	0	0	0	0	0	0	0
V20	5	7	8	0	0	5	0	0	14	0	14	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V21	0	0	0	25	0	4	0	4	0	5	0	10	0	0	0	0	0	0	0	0	0	0	0	0	0
V22	4	0	4	0	4	0	4	0	0	10	0	0	11	0	0	0	0	0	0	0	0	0	0	0	0
V23	4	4	6	0	0	2	0	4	7	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V24	0	0	0	0	0	12	6	2	0	0	0	7	0	10	8	11	7	0	0	0	0	0	0	0	0
V25	0	0	0	0	0	4	6	3	7	1	0	1	8	10	5	0	4	4	3	0	2	0	0	0	0
V26	0	0	0	0	0	4	4	2	7	3	0	7	4	6	0	2	2	0	14	5	0	0	0	0	0
V27	0	0	0	0	0	7	5	0	3	0	10	0	20	0	0	5	0	5	0	0	3	1	0	0	0
V28	0	0	0	0	0	10	0	5	0	0	0	5	0	0	14	0	0	0	0	6	0	0	0	0	0
V29	0	0	0	0	0	0	5	0	3	0	7	0	0	10	0	0	3	0	7	0	0	0	0	0	0
V30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V31	0	0	0	0	0	0	0	0	0	0	0	0	5	4	3	0	4	0	7	0	12	8	0	6	0
V32	0	0	0	0	0	0	0	0	0	0	0	0	3	5	10	0	0	8	0	6	0	8	0	0	6

V33	0	0	0	0	0	0	0	0	0	0	0	0	3	4	6	4	7	0	11	0	8	0	6	3	4
V34	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	26	0	5	0	7	0	8	0
V35	0	0	0	0	0	0	0	0	0	0	0	0	7	0	3	0	10	0	7	6	8	0	6	0	4
V36	0	0	0	0	0	0	0	0	0	0	0	0	6	0	4	0	0	12	0	4	0	0	6	3	0
V37	0	0	0	0	0	0	0	0	0	0	0	0	0	10	4	0	0	7	0	0	6	0	6	10	0
V38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	7	0	3	4	7
V39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	4	0	7	1	6
V40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	5	5	0	7	0
V41	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	0	7	6	4	0
V42	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	7	8	6	2	6
V43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	7	5	4	13	0
V44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	0	0	4

Tabla 26: Matriz flujo de pasajeros para la comparación. Los elementos que no se muestran en la tabla son todos 0.

Con toda esta información introducida en los distintos programas se obtienen las siguientes asignaciones.

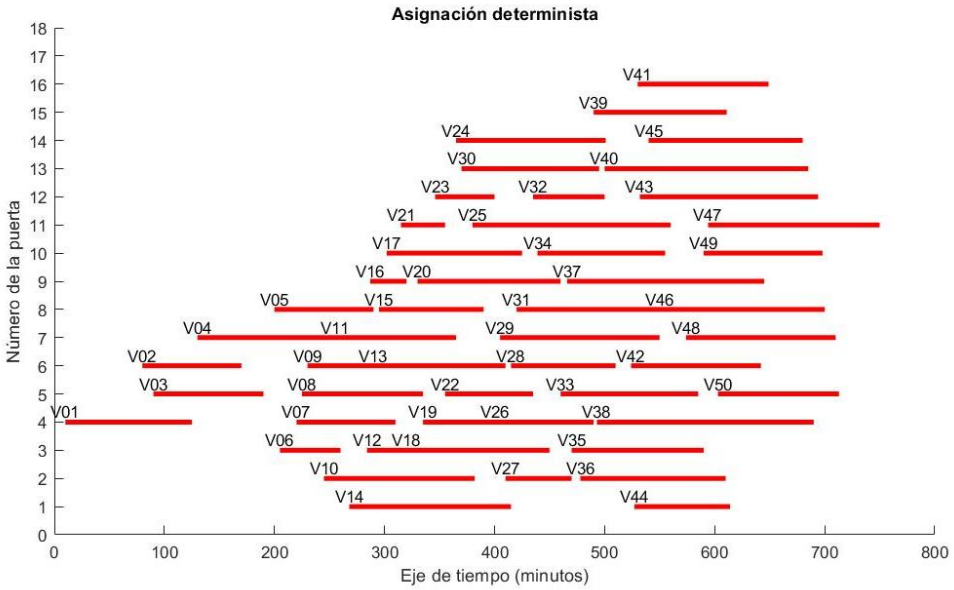


Imagen 3: Resultado de la asignación determinista

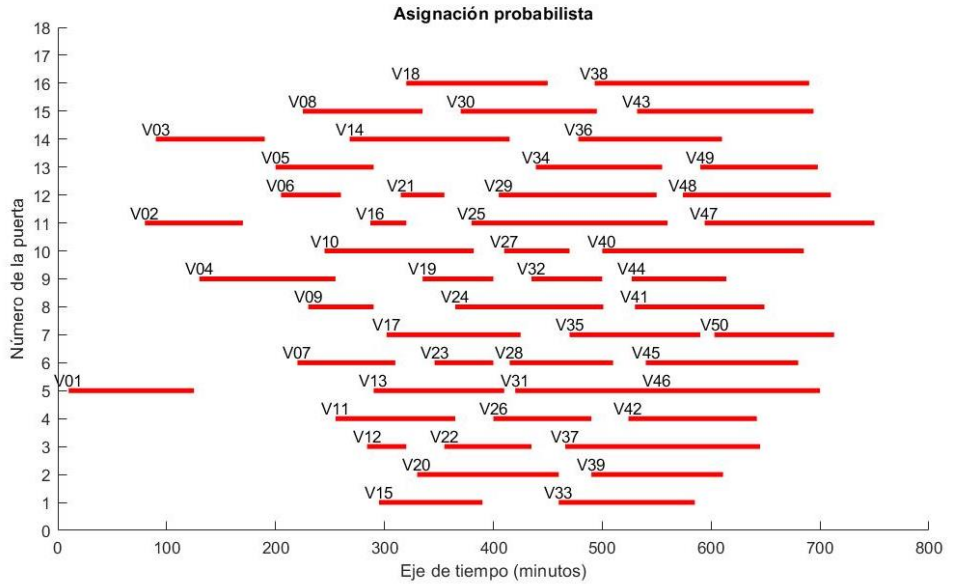


Imagen 4: Resultado de la asignación probabilista

Las imágenes muestran el eje de tiempos de cada una de las puertas, marcando el tiempo que, de media, está la aeronave en la puerta. Arriba de la hora de llegada de cada vuelo se muestra su identificación. Como se observa fácilmente, el caso determinista tiene varios casos en los que hay puntos muy próximos, lo cual significa que el riesgo de conflicto en ese determinado instante es potencialmente alto. De forma contraria, apenas se observa un caso en el que los puntos están muy próximos en la asignación probabilista (V31 con V46) de forma que quedan mucho más espaciados entre sí. Analizando los datos numéricos de cada una de las simulaciones se pueden extraer las siguientes conclusiones:

	Asignación determinista	Asignación probabilista
Distancia total recorrida por los pasajeros	271 140m	325 140 m
Probabilidad de conflicto global	-	9.206
Promedio del tiempo entre vuelos	24.29 minutos	48.03 minutos

Tabla 27: Resultado de la comparación

Se observa aquí que el tiempo medio entre vuelos asignados a una misma puerta aumenta en un 98% al pasar de la asignación determinista a la probabilista. No obstante, el precio que se deberá pagar por esta robustez añadida al sistema es el incremento en 54 000 m de distancia a recorrer por los pasajeros. Para poner esta cifra en contexto resulta útil observar que equivale a 22.82 m "extra" para cada uno de los 2366 pasajeros que realizan un transbordo en la simulación estudiada. Esto significa que cada pasajero recorrerá una distancia que no llega a la que, de media, existe entre puertas (33 m). Se debe tener en cuenta que, para este caso, la suma del promedio de las varianzas en la hora de llegada y salida es de 20 minutos, por lo que resultará bastante probable encontrar algún conflicto si se utiliza la asignación determinista. Una vez sabido esto, corresponde al operador aeroportuario decidir qué tipo de asignación le resulta más conveniente en función de la disponibilidad de slots alejados y la facilidad para trasladar a los pasajeros desde estos. No obstante, para esta simulación en concreto parece mucho más oportuno utilizar una asignación probabilista ya que la distancia extra, si se reparte entre todos los pasajeros, es pequeña; al mismo tiempo, la robustez añadida es considerable.

Capítulo 8
Presupuesto

El presupuesto utilizado en la elaboración del proyecto de puede descomponer en tres secciones que se pueden desglosar de forma separada.

Coste de personal

Esta es la parte que más recursos necesita ya que han sido muchas las horas invertidas en el proyecto. Se estima que el coste por hora trabajada es de 20€ para el estudiante y 40€ para el profesor tutor. Aunque la duración del proyecto ha sido de 5 meses, no en todos se ha dedicado el tiempo completo al mismo. Así, se establece el número total de horas trabajadas en 320. Aplicando el precio por hora establecido, el trabajo del estudiante supone un coste de 6400€. Por otra parte, el profesor tutor ha asesorado al alumno en un total de 7 reuniones con una duración aproximada de una hora y media. Esto es un total de 10.5 horas de trabajo que equivalen a un coste de 420€. Queda por tanto el coste de personal definido en 6820€. De forma resumida se puede observar el desglose en la Tabla 28.

Concepto	*Precio por unidad (€)*	*Cantidad (h)*	*Precio total (€)*
Coste laboral Alumno	20	320	6400
Coste laboral Tutor	40	10.5	420
Coste de personal total			**6820**

Tabla 28: Desglose de costes de personal

Coste de equipos y software

Para la elaboración del proyecto se ha utilizado principalmente un ordenador equipado con el software específico que ha permitido realizar las simulaciones, así como el software de ofimática utilizado en la redacción del informe. De forma ordenada por horas de uso se han utilizado los siguientes componentes:

- Equipo ordenador portátil Asus UX550V con licencia Windows 10 Home. Coste total del equipo en el año de la compra (2018) 1400€. Suponiendo una depreciación anual del 26% se encuentra que al principio del proyecto su valor era de 528 €. Así, tendiendo en cuenta que el proyecto ha durado 5 meses, se fija el coste del equipo en 260€.
- Software de programación lineal GAMS. Es el software utilizado para todas las simulaciones realizadas en el proyecto. La licencia base del programa tiene un precio de 640$ para instituciones educativas a la que se le debe sumar el coste del algoritmo de resolución utilizado (Antigone) cuyo coste es de unos 2560$ adicionales. Así, solo el

software de simulación de programación lineal supone un coste de 3200$ que equivale a 2679.92€. Pese a que la licencia es de por vida, se asume que cada cinco años se tendrá que renovar la versión porque se habrá quedado obsoleta. Así, repartiendo el coste se tiene que la utilización de los cinco meses equivale a un coste de 223.33€

- Software de ofimática utilizado en la redacción del informe (Microsoft Word), elaboración de gráficos (Microsoft Power Point) y tratamiento de datos (Microsoft Excel). El coste anual de una licencia de Microsoft Office que incluye todas las herramientas anteriores es de 69€ anuales. Asumiendo una utilización de 5 meses, el coste para el proyecto ha sido de 28.75€.
- Software de programación (Matlab) utilizado para elaborar los datos de entrada en la fase 2 del estudio probabilista. El coste anual de la licencia para instituciones educativas es de 250€. Al haberse utilizado durante un mes se determina un coste para el proyecto de 20.83€.
- Software matemático utilizado en la categorización de las funciones de probabilidad (Wolfram Mathematica). El coste total de la licencia anual para organizaciones educativas es de 1590€ y, como se ha utilizado solo durante medio mes, se asume un coste para el proyecto de 66.25€.

Todo esto queda resumido en la Tabla 29:

Herramienta	*Precio licencia*	*Periodo utilización*	*Coste para el proyecto*
Asus UX550V con Windows 10 Home		5 meses	260 €
GAMS	2679.92€	5 meses (renovación licencia cada 5 años)	223.33€
Ofimática Microsoft Office	69€	5 meses	28.75€
Matlab	250€	1 mes	20.83€
Wolfram Mathematica	1590€	0.5 meses	66.25€
Total coste equipos y software			**627.91€**

Tabla 29: Desglose de costes de equipos y software

Costes indirectos

Se incluyen en esta categoría los costes que no dependen directamente del proyecto pero que se generan mientras se lleva a cabo. Se incluyen aquí, por tanto, los costes del trabajo de personal de la administración, servicios que presta la universidad al alumnado, así como los gastos por el uso de la infraestructura. Dado que se trata de costes de muy variado tipo y

difíciles de estimar, se ha presupuestado como un 5% del total del presupuesto. Con esta suposición queda un total en costes indirectos de 372.40€.

Así, se puede calcular el coste total del proyecto sumando todas las partes como se ve en la Tabla 30:

Concepto	*Total (€)*
Coste Personal	6820
Coste Equipo	627.91
Costes indirectos	372.40
Coste total	**7820.31**

Tabla 30: Presupuesto total del proyecto

Capítulo 9
Conclusiones

A lo largo de la memoria se ha visto que el problema está lejos de resolverse por completo ya que, por una parte, existen multitud de criterios distintos en función del interesado (aerolíneas, gestor aeroportuario…), y por otra, se trata de problemas NP-hard para los que encontrar una solución completa resulta imposible. No obstante, se han estudiado los distintos métodos propuestos a lo largo de los años y se han diseñado dos sistemas propios que proponen nuevas formas de afrontar el problema, sobre todo, en la parte probabilista. La principal ventaja de los sistemas propuestos es que se pueden resolver con independencia del algoritmo de resolución, estando definidos de forma que cualquier método de programación lineal los puede interpretar.

Como punto final del proyecto se ha encontrado que las soluciones probabilistas en dos fases, como la propuesta, podrían resultar más beneficiosas que las deterministas, porque los pasajeros apenas notarían la diferencia de distancia con respecto a uno óptimo en este sentido. No obstante, no se han podido establecer conclusiones generales ya que, según las características particulares de cada aeropuerto en cuanto a configuración de puertas o influencia de los retrasos, será más eficiente un sistema u otro. De la misma forma, es importante resaltar que los datos de las simulaciones utilizados en el proyecto han sido generados de forma aleatoria y, por tanto, podrían no tener en cuenta algunas cuestiones que sí se dan en la realidad. Antes de utilizar alguno de los modelos propuestos sería conveniente validarlos previamente con datos reales.

Igualmente, se deben reconocer las limitaciones del proyecto, como la falta de profundización en el funcionamiento interno de los algoritmos de resolución. Dado que la programación lineal no se ha trabajado durante el grado, se han tenido que investigar los aspectos clave para el proyecto sin tener un conocimiento global en la materia. Lo mismo ha sucedido con el software de programación lineal, GAMS, que se ha aprendido mientras se desarrollaban los modelos. No obstante, gracias a su sencillez y a la similitud de su código con el matemático, se ha podido utilizar de forma satisfactoria.

De cara a futuros estudios sobre la optimización de las puertas de embarque se podrían estudiar otras cuestiones que pueden afectar a las operaciones de los aviones. Esto es, incluir restricciones al modelo para que se tengan en consideración los siguientes aspectos:

- Distintas distribuciones de las puertas en la terminal. Dada la forma de modelización de la distancia entre puertas, tan solo se puede utilizar con terminales lineales. Para algunas configuraciones distintas, como terminales satélites o paralelas, el modelo no consideraría correctamente las distancias.
- Las limitaciones en el push-back de las aeronaves. El push-back es el procedimiento mediante el que se remolca a la aeronave desde la puerta de embarque hasta la calle de rodaje. Para esta operación se debe utilizar un tractor de remolque que debe estar disponible en la ubicación de la aeronave en el momento de su salida y, si no se tiene,

se deberá retrasar la salida. Desde este punto de vista, si el tractor debe remolcar a dos aeronaves en poco tiempo, estas no deberían estar situadas muy lejos entre sí.

- El tráfico en las calles de rodaje. El procedimiento de push-back requiere de cierta precisión, se debe hacer despacio y, por tanto, se bloquea la calle de rodaje mientras se realiza la maniobra. Esto significa que otra aeronave que deba pasar por la calle o que quiera estacionar en una puerta adyacente, deberá esperar. De esta forma, las aeronaves con horas de llegada y salida parecidas deberían estar ligeramente separadas para no interferir.
- Agrupación de vuelos por aerolíneas. Generalmente, las aerolíneas suelen tener a personal encargado de asistir a los pasajeros en su subida al avión. Así, les interesará que sus vuelos estén cerca para que este personal se pueda mover fácilmente entre las puertas en las que opera la aerolínea.

Referencias

[1] Wikipedia. (mayo de 2021). Obtenido de https://es.wikipedia.org/wiki/Aeropuerto_Adolfo_Su%C3%A1rez_Madrid-Barajas#:~:text=Dispone%20de%2020%20puertas%20de,cintas%20de%20equipaje%20cada%20una

[2] Data.Worldbank. (mayo de 2021). Obtenido de https://data.worldbank.org/indicator/IS.AIR.PSGR?end=2019&start=1970&view=chart

[3] Data.Worldbank. (mayo de 2021). Obtenido de https://data.worldbank.org/indicator/IS.AIR.GOOD.MT.K1?end=2019&start=1970&view=chart

[4] *Estos son los aeropuertos más y menos puntuales de España*. (2019). *Traveler.es*. Obtenido de https://www.traveler.es/viajeros/articulos/aeropuertos-mas-puntuales-de-espana-en-2018-segun-airhelp/14769

[5] Ciesluk, K. (2020). Turnaround Time: Why It's Important And How Airlines Can Speed It Up. *Simple Flying*. Obtenido en https://simpleflying.com/turnaround-time-importance/

[6] Bazargan, M. (2010). Airline operations and scheduling. Ashgate Publishing Limited.

[7] Thanyan AL-Sultan, A. (2012) The Airport Gate Assignment Problem: Scheduling. Algorithms and Simulation Approach. Okayama University.

[8] Gürsoy, M., & Akyildlz Alçura, G. (February 2015). A NEW GATE ASSIGNMENT POLICY AND THE EFFECT OF GATE OCCUPANCY TIME ON THE LEVEL OF SERVICE AT AIRPORT TERMINALS. Indoor Air.

[9] Chapter four: Solving the gate assignment problem. Utrecht University - Department of Information and Computing Sciences. Scheduling and Timetabling course (INFOSTT), http://www.cs.uu.nl/docs/vakken/stt/Chapter-gates.pdf, Visitado el 18-06-2020.

[10] Ding, H., Lim, A., Rodrigues, B. & Zhu, Y. (s.f.). The Airport Gate Assignment Problem. National University of Singapore. Hong Kong University of Science and Technology. Singapore Management University.

[11] Schaijk, O. R., & Visser, H. G. (2017). Robust flight-to-gate assignment using flight presence probabilities. Transportation Planning and Technology.

[12] Castaing, J., Mukherjee, I., Cohn, A., Hurwitz, L., Nguyen, A. & Müller, J. J. (2014). Reducing airport gate blockage in passenger aviation: Models and analysis. Industrial and Operations Engineering Department, University of Michigan.

[13] Pesch, E., Dorndorf, U., & Jaehn, F. (s.f.). Flight Gate Allocation: Modells, Methods and Robust solutions.

[14] Castillo, E., Conejo, A., Pedregal, P., García, R., & Alguacil, N. (2002). Formulación y resolución de modelos de programación matemática en ingeniería y ciencia. Departamento de Ingeniería Artificial. ETS ingenieros informáticos. UPM. Disponible en http://www.dia.fi.upm.es/~jafernan/teaching/operational-research/LibroCompleto.pdf

[15] Franco Gálvez, M. A. (2015). Técnicas heurísticas y metaheurísticas para el problema de la máxima diversidad (Maximum Diversity Problem (MDP)). Facultad de Informática. Universidad de Murcia.

[16] GAMS Development Corp. & GAMS Software GmbH. (Mayo de 2021). *GAMS Documentation Center*. Obtenido de https://www.gams.com/latest/docs/index.html

Anexos

Código de Modelo básico: asignación de pasajero a puerta destino

```
$title Problema base: Mínima distancia recorrida por los pasajeros

Set
   i 'vuelo de origen' / F1*F7 /
   j 'puerta de origen'
        /Puerta3,Puerta4,Puerta10,Puerta11,Puerta14,Puerta15,Puerta17/
   k 'puerta de destino' /Puerta1*Puerta19/;


Table dP(j,k) 'Distancia entre puertas'
            Puerta1    Puerta2 Puerta3 Puerta4 Puerta5 Puerta6 Puerta7 Puerta8 Puerta9
Puerta10    Puerta11    Puerta12    Puerta13    Puerta14    Puerta15    Puerta16    Puerta17
Puerta18    Puerta19
      Puerta3     10       40      0       30      10      40      20      50      30      60          40
70          50          80          60          90          70          90          80
      Puerta4     40       10      30      0       40      10      50      20      60      30          70
40          80          50          90          60          90          70          80
      Puerta10    70       40      60      30      50      20      40      10      30      0           40
10          50          40          60          30          70          40          50
      Puerta11    50       80      40      70      30      60      20      50      10      40          0
30          10          40          20          50          30          50          40
      Puerta14    90       60      80      50      70      40      60      30      50      20          40
10          30          0           40          10          50          20          30
      Puerta15    70       100     60      90      50      80      40      70      30      60          20
50          10          40          0           30          10          30          20
      Puerta17    80       100     70      90      60      80      50      70      40      60          30
50          20          40          10          30          0           20          10;


Table P(i,k) 'Número de pasajeros'
            Puerta1    Puerta2 Puerta3 Puerta4 Puerta5 Puerta6 Puerta7 Puerta8 Puerta9
Puerta10    Puerta11    Puerta12    Puerta13    Puerta14    Puerta15    Puerta16    Puerta17
Puerta18    Puerta19
      F1          5       5       10      8       15      8       2       10      8       20          5           4
0           9           3           4           1           2           1
      F2          5       2       1       4       19      9       4       2       3       2           27          3
8           4           0           2           1           7           2
      F3          10      0       4       9       13      4       4       4       3       5           5           8
4           9           11          7           9           4           4
      F4          4       8       5       4       10      4       1       0       0       2           4           19
1           2           4           5           5           8           2
      F5          4       11      9       9       6       3       1       4       4       2           1           0
3           5           1           2           2           3           4
      F6          1       2       42      5       2       7       6       2       4       7           2           3
6           4           10          2           1           0           0
```

```
      F7      3      3     2    5      9    13     11       2     2     3     7          22
4       0       1       1       2       2       9;

Binary Variable x(i,j) 'variable binaria 1 si vuelo i está asignado a puerta j';

Variable z 'Distancia total';

Equation
   distancia      'Función objetivo'
   solo_una_puerta(i) 'Cada avión estará aisgnado a una sola puerta'
   solo_un_vuelo(j) 'Cada puerta albergara a solo un vuelo';

distancia..      z =e= sum(i,sum(j,sum(k,P(i,k)*dP(j,k)*x(i,j))));

solo_una_puerta(i).. sum(j, x(i,j)) =e= 1;

solo_un_vuelo(j).. sum(i, x(i,j)) =l= 1;

Model transport / all /;

solve transport using mip minimizing z;

display z.l, x.l, x.m;
```

Código de Mejora 1: Asignación de pasajero a vuelo de destino

```
$title Problema desarrollado totalmente de cero: separación en pasajeros que aporta y pasajeros que recibe.

$onText
Modificación del original para que el vuelo de salida y el de llegada tengan la misma puerta si son V1
$offText

Set
  i 'Vuelo en el que llega el pasajero' /V1*V3/
  j 'Vuelo en el que se va el pasajero' /V1*V3/
  puert 'puerta de embarque' /P1*P3/;
Table P(i,j) 'Pasajeros de cada vuelo'
            V1  V2  V3
      V1    00  50  30
      V2    00  00  01
      V3    00  00  00 ;
Parameters
   dP(puert)  Distancia desde origen hasta P3
   /  P1  1000
      P2  800
      P3  0   /;
Binary Variable x(i,puert) 'variable binaria 1 si vuelo i está asignado a puerta k';
Binary Variable y(j,puert) 'variable binaria 1 si vuelo j está asignado a puerta l';
Variable z 'Desplazamiento total';

Equation
  distancia      'Función objetivo'
  origen_max1p(i) 'Cada vuelo de origen una puerta'
  destino_max1p(j) 'Cada vuelo de destino una puerta'
  origen_1vxpuerta(puert)  'Max 1 vuelo por cada puerta origen'
  destino_1vxpuerta(puert) 'Max 1 vuelo por cada puerta destino'
  mismo_vuelo1(puert)    'El vuelo de llegada y el de salida deben ser el mismo'
  mismo_vuelo2(puert)    'El vuelo de llegada y el de salida deben ser el mismo'
  mismo_vuelo3(puert)    'El vuelo de llegada y el de salida deben ser el mismo';

distancia..    z =e= sum(i,sum(j,P(i,j)*abs(sum(puert,dP(puert)*x(i,puert))-
sum(puert,dP(puert)*y(j,puert)))));
origen_max1p(i).. sum(puert, x(i,puert)) =e= 1;
destino_max1p(j).. sum(puert,y(j,puert)) =e= 1;
origen_1vxpuerta(puert)..   sum(i,x(i,puert)) =l= 1;
destino_1vxpuerta(puert)..  sum(j,y(j,puert)) =l= 1;

mismo_vuelo1(puert)..     x('V1',puert) =e= y('V1',puert);
mismo_vuelo2(puert)..     x('V2',puert) =e= y('V2',puert);
mismo_vuelo3(puert)..     x('V3',puert) =e= y('V3',puert);

Model transport / all /;
Option MINLP = antigone;
```

```
solve transport using minlp minimizing z;
display z.l, x.l, y.l;
```

Código de Mejora 2: Añadir horarios de llegada y salida

```
$title Problema desarrollado totalmente de cero. Inclusión de horas de llegada y salida
$onText
Modificación del caso propio para que identifique los tiempos de ocupación de la puerta.
IMPORTANTE: no se pueden poner dos vuelos con mismo valor de ai. Si de da el caso,
utilícense decimales.
$offText

Set
   i 'Vuelo en el que llega el pasajero' /V1*V5/
   j 'Vuelo en el que se va el pasajero' /V1*V5/
   puert 'puerta de embarque' /P1*P3/;
Table
P(i,j) 'Pasajeros de cada vuelo'
                V1  V2  V3  V4  V5
          V1    00  50  30  50  30
          V2    00  00  00  00  00
          V3    00  00  00  00  00
          V4    00  00  00  00  00
          V5    00  00  00  00  00;

Parameters
   dP(puert)  Distancia desde origen hasta P3
   /  P1  1000
      P2  800
      P3  0   /
   ai(i)    Hora de llegada del vuelo
   /  V1   0
      V2   10
      V3   15
      V4   50
      V5   55  /

   di(i)    Hora de salida del vuelo
   /  V1  40
      V2  50
      V3  55
      V4  120
      V5  120  /
   aj(j)    Hora de llegada del vuelo
   /  V1   0
      V2   10
      V3   15
      V4   50
      V5   55  /

   dj(j)    Hora de salida del vuelo
   /  V1  40
      V2  50
      V3  55
```

```
          V4  120
          V5  120  /;


    Binary Variable x(i,puert) 'variable binaria 1 si vuelo i está asignado a puerta k';
    Binary Variable y(j,puert) 'variable binaria 1 si vuelo j está asignado a puerta l';
    Variable z 'Desplazamiento total';

    Equation
      distancia      'Función objetivo'
      origen_max1p(i) 'Cada vuelo de origen una puerta'
      destino_max1p(j) 'Cada vuelo de destino una puerta'
      origen_1vxpuerta(puert,i,j)  'Max 1 vuelo por cada puerta origen si coinciden en espacio
temporal'
      mismo_vuelo1(puert)    'El vuelo de llegada y el de salida deben ser el mismo'
      mismo_vuelo2(puert)    'El vuelo de llegada y el de salida deben ser el mismo'
      mismo_vuelo3(puert)    'El vuelo de llegada y el de salida deben ser el mismo'
      mismo_vuelo4(puert)    'El vuelo de llegada y el de salida deben ser el mismo'
      mismo_vuelo5(puert)    'El vuelo de llegada y el de salida deben ser el mismo';

    distancia..    z =e= sum(i,sum(j,P(i,j)*abs(sum(puert,dP(puert)*x(i,puert))-
sum(puert,dP(puert)*y(j,puert)))));

    origen_max1p(i).. sum(puert, x(i,puert)) =e= 1;
    destino_max1p(j).. sum(puert,y(j,puert)) =e= 1;

    origen_1vxpuerta(puert,i,j).. x(i,puert)*y(j,puert)*abs(ai(i)-aj(j))*(dj(j)-ai(i))*(di(i)-aj(j)) =l=
0;

    mismo_vuelo1(puert)..     x('V1',puert) =e= y('V1',puert);
    mismo_vuelo2(puert)..     x('V2',puert) =e= y('V2',puert);
    mismo_vuelo3(puert)..     x('V3',puert) =e= y('V3',puert);
    mismo_vuelo4(puert)..     x('V4',puert) =e= y('V4',puert);
    mismo_vuelo5(puert)..     x('V5',puert) =e= y('V5',puert);

    Model transport / all /;
    Option MINLP = antigone;
    transport.optfile=1;
    solve transport using minlp minimizing z;
    display z.l, x.l, y.l;
```

Código de Fase 1: Mínimo riesgo de conflicto global

```
$title Problema desarrollado totalmente de cero. Introducción al método probabilista
$onText
Creación de un modelo que asigna vuelos a puertas para un mínimo riesgo de conflicto
global
$offText

$FuncLibIn stolib stodclib

Function pdfNormal    /stolib.pdfNormal/;
Function cdfNormal    /stolib.cdfNormal/;

Set
   i 'Vuelos que deben ser atendidos en el aeropuerto' /V1*V5/
   j 'Vuelos que deben ser atendidos en el aeropuerto' /V1*V5/
   hor 'Horario de puerta' /H1*H3/
   minute  'minutos de simulación'/1*350/;
Parameters
    mediaa(i)  Media función de probabilidad de la llegada (arrival)
    /  V1  90
       V2  110
       V3  125
       V4  160
       V5  205/

    desva(i) desviación estándar de función de probabilidad de la llegada (arrival)
    /  V1  5
       V2  10
       V3  20
       V4  20
       V5  15/

    mediad (i)  Media función de probabilidad de la salida (departure)
    /  V1  120
       V2  230
       V3  245
       V4  250
       V5  300/

    desvd (i)   Desviacion estandar de funcion de probabilidad de salida (departure)
    /  V1  5
       V2  10
       V3  10
       V4  30
       V5  20/;

    parameter
    probi(minute,i) Definicion eq funcion de probabilidad;
```

```
    probi(minute,i) = min( cdfNormal(ord(minute),mediaa(i),desva(i)), 1-
cdfNormal(ord(minute),mediad(i),desvd(i)));
        parameter
        probj(minute,j) Definicion eq funcion de probabilidad para j;

    probj(minute,'V1') = probi(minute,'V1');
    probj(minute,'V2') = probi(minute,'V2');
    probj(minute,'V3') = probi(minute,'V3');
    probj(minute,'V4') = probi(minute,'V4');
    probj(minute,'V5') = probi(minute,'V5');

    Binary Variable x(i,hor) 'variable binaria 1 si vuelo i está asignado al horario hor';
    Binary Variable y(j,hor) 'variable binaria 1 si vuelo j está asignado al horario hor';

    Variable p 'Riesgo de conflicto global';

    Equation
      conflicto      'Función objetivo'
      max1h(i) 'Cada vuelo de origen un horario'
      mismovuelo1(hor)  'Los vuelos i y j deben estar asignados al mismo horario'
      mismovuelo2(hor)  'Los vuelos i y j deben estar asignados al mismo horario'
      mismovuelo3(hor)  'Los vuelos i y j deben estar asignados al mismo horario'
      mismovuelo4(hor)  'Los vuelos i y j deben estar asignados al mismo horario'
      mismovuelo5(hor)  'Los vuelos i y j deben estar asignados al mismo horario';

    conflicto..    p =e=sum(minute,sum(hor,sum(i,sum(j$(ord(j) > ord(i)),
(x(i,hor)*probi(minute,i)*y(j,hor)*probj(minute,j))))));

    max1h(i).. sum(hor, x(i,hor)) =e= 1;

    mismovuelo1(hor).. x('V1',hor) =e= y('V1',hor);
    mismovuelo2(hor).. x('V2',hor) =e= y('V2',hor);
    mismovuelo3(hor).. x('V3',hor) =e= y('V3',hor);
    mismovuelo4(hor).. x('V4',hor) =e= y('V4',hor);
    mismovuelo5(hor).. x('V5',hor) =e= y('V5',hor);

    Model transport / all /;
    Option MINLP = antigone;
    solve transport using minlp minimizing p;
    display p.l, x.l, y.l;
```

Código de Fase 2: Mínima distancia andada por los pasajeros

```
$title Problema desarrollado totalmente de cero. Método probabilista fase dos
$onText
Creación de un modelo que asigna horarios a puertas para que la distancia que caminen los
psajeros sea mínima.
$offText

Set
   i_hor 'Horario de origen' /H1*H3/
   j_hor    'Horario de destino' /H1*H3/
   puert 'puerta de embarque' /P1*P3/;
Table
P(i_hor,j_hor) 'Pasajeros de cada horario'
            H1  H2  H3
      H1  0   0   0
      H2  80  50  30
      H3  0   0   0;
Parameters
    dP(puert)  Distancia desde origen hasta P3
   /  P1  1000
      P2  800
      P3  0   /;

Binary Variable x(i_hor,puert) 'variable binaria 1 si horario i_hor está asignado a la puerta
puert';
Binary Variable y(j_hor,puert) 'variable binaria 1 si horario j_hor está asignado a la puerta
puert';
Variable z 'Desplazamiento total';

Equation
   distancia      'Función objetivo'
   max1p(i_hor) 'Cada horario una puerta'
   max_1hxpuerta(puert)  'Max 1 horario por cada puerta'
   mismovuelo1(puert)  'Los horarios i_hor y j_hor deben estar asignados a la misma puerta'
   mismovuelo2(puert)  'Los horarios i_hor y j_hor deben estar asignados a la misma puerta'
   mismovuelo3(puert)  'Los horarios i_hor y j_hor deben estar asignados a la misma puerta';

distancia..    z =e=
sum(i_hor,sum(j_hor,P(i_hor,j_hor)*abs(sum(puert,dP(puert)*x(i_hor,puert))-
sum(puert,dP(puert)*y(j_hor,puert)))));
max1p(i_hor).. sum(puert, x(i_hor,puert)) =e= 1;
max_1hxpuerta(puert).. sum(i_hor, x(i_hor,puert)) =l= 1;

mismovuelo1(puert).. x('H1',puert) =e= y('H1',puert);
mismovuelo2(puert).. x('H2',puert) =e= y('H2',puert);
mismovuelo3(puert).. x('H3',puert) =e= y('H3',puert);

Model transport / all /;
Option MINLP = antigone;
solve transport using minlp minimizing z;
```

```
display z.l, x.l, y.l;
```

Código matlab para pasar de fase 1 fase 2 en la asignación probabilista

```
   datos=readtable('datos.csv'); %Matriz pasajeros entre vuelos
   datos(:,1)=[];
   sol=readtable('sol.csv'); %Matriz solución de asignación en fase 1

   sol(:,1)=[];

   Hsol=zeros(width(sol));

   for datos_filas=1:height(datos)
       for datos_columnas=1:width(datos)
           for k=1:width(sol)
               if sol{datos_filas,k}==1
                   Hsol_filas=k;
               end
           end
           for m=1:width(sol)
               if sol{datos_columnas,m}==1
                   Hsol_columnas=m;
               end
           end

Hsol(Hsol_filas,Hsol_columnas)=Hsol(Hsol_filas,Hsol_columnas)+datos{datos_fi
las,datos_columnas};
       end
   end
```